Adaptive LLM Routing for Multi-Turn Conversations with Continuously Evolving User Queries

Jiarui Zhang[1] **Xiangyu Liu**[2] **Yong Hu**[2] **Chaoyue Niu**[1*] **Hang Zeng**[1]
Shaojie Tang[3] **Fan Wu**[1] **Guihai Chen**[1]

[1]Shanghai Jiao Tong University, Shanghai, China
[2]WeChat, Tencent Inc, Beijing, China
[3]University at Buffalo, New York, United States

Abstract

Multi-turn conversation is the predominant form of interaction with large language models (LLMs), where user queries continuously evolve. However, existing LLM routing methods are primarily designed for single-turn interactions or fixed queries settings, overlooking the dynamic nature of multi-turn conversations and the challenge of delayed rewards, thereby limiting their ability to optimize cumulative performance. To address this challenge, we move from myopic, single-turn selection to long-horizon routing for multi-turn conversation. Accordingly, we propose ConvoRouter, which first performs MCTS to explore conversation branches induced by different LLM selections and collect trajectories with high cumulative rewards. ConvoRouter then learns a lightweight routing policy from search-derived data, augmented with retrieval-based future state approximation, enabling multi-turn routing without online search. Experiments on both open-domain and domain-specific conversation tasks across diverse candidate sets of both open-source and closed-source LLMs demonstrate that ConvoRouter significantly outperforms single LLMs and existing routing baselines in task success rate, while achieving a superior performance-cost trade-off when combined with a cost-aware reward. [2]

1 Introduction

Multi-turn conversation is the dominant paradigm of user interaction with large language models (LLMs). In everyday general-purpose applications such as ChatGPT [1], Gemini [8], Qwen [37] and DeepSeek [21], user engagement rarely consists of isolated queries; instead, users naturally participate in iterative conversations, with continuously evolving queries, providing clarification, feedback, and refinements as tasks progress [38]. This multi-turn interaction pattern is also evident across diverse LLM-driven vertical applications, including e-commerce customer service [19], medical consultation [41], educational tutoring [22], emotional support [5], and programming assistance [28].

Multi-turn conversations encompass diverse user intents that impose varying demands on LLM capabilities. Meanwhile, different LLMs exhibit complementary strengths in instruction following, knowledge coverage, and reasoning [20]. To leverage this complementarity, LLM routing selects the most suitable LLM for each query from a pool of pre-deployed LLMs [6] and has been shown to outperform strong individual LLMs [17, 4, 42]. However, most existing routing methods are developed and evaluated for single-turn interactions or settings with fixed queries, even when extended to multi-step reasoning or tool-use scenarios within a given task. In contrast, multi-turn conversations involve continuously evolving user queries and contexts. This gap raises a natural

*Chaoyue Niu is the corresponding author (rvince@sjtu.edu.cn).
[2]The code and data are available at `https://github.com/Oww099/ConvoRouter`.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

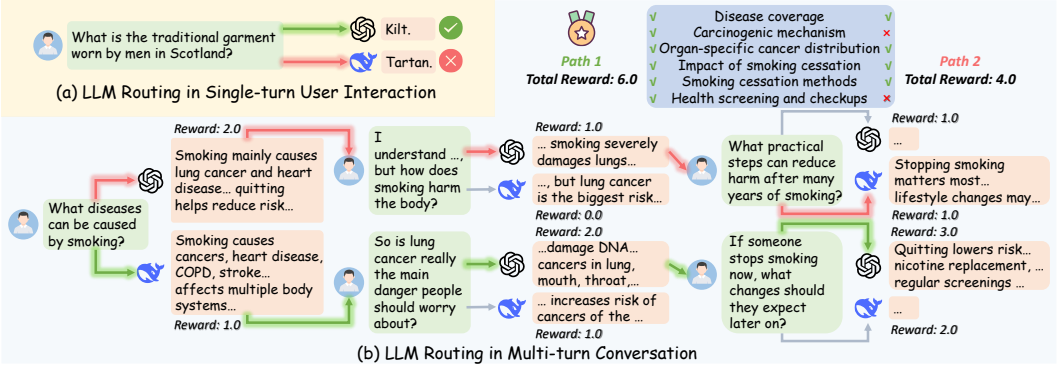


Figure 1: Illustration of LLM routing in single-turn interaction (a) and multi-turn conversation (b). Green arrows indicate ideal decisions. In multi-turn conversation, user queries evolve dynamically. Rewards are distributed across turns, while the total reward reflects overall user intent fulfillment.

question: *How should LLM routing be designed for multi-turn conversations with continuously evolving user queries, where decisions are no longer independent across turns?*

As illustrated in Figure 1(a), in single-turn user interaction, users seek a correct answer to a clearly defined query, and the router simply selects the best-performing LLM. In contrast, in multi-turn conversation, users progressively advance a set of interrelated core intents across turns, with success defined by whether these intents are collectively fulfilled throughout the conversation. As shown in Figure 1(b), multi-turn conversation routing is an evolving decision process, where the router selects a LLM at each turn, receiving immediate rewards for intent progression and a total reward reflecting overall intent fulfillment. Notably, selecting different LLMs under the same context can elicit different follow-up user queries in subsequent turns, causing the conversation to evolve along distinct trajectories and highlighting its **dynamic** nature. Moreover, overall intent fulfillment can only be assessed after the conversation unfolds, exhibiting clear **delayed rewards**.

Due to the dynamic and delayed-reward nature of multi-turn conversation, per-turn observable signals often fail to capture the long-term impact of LLM selection. As illustrated in Figure 1(b), a locally optimal choice at a given turn does not necessarily yield a globally optimal conversation outcome. Beyond performance, inference costs accumulate over turns, and frequent model switching disrupts cache reuse, making locally cost-optimal routing suboptimal at the conversation level. We thus argue for a shift from myopic, per-turn selection to long-horizon LLM routing in multi-turn conversation.

In this work, we propose **ConvoRouter**, a LLM routing framework for multi-turn conversation. We first employ Monte Carlo Tree Search (MCTS) with a user simulator to explicitly explore conversation branches induced by different LLM selections. By rolling out complete conversation trajectories and evaluating their cumulative returns with a reward model, we collect routing trajectories with high long-term returns. The reward model adopts a checklist-based formulation, well suited to fine-grained user intent fulfillment in multi-turn conversation. Offline policy learning is then performed on these trajectories via behavior cloning, as online search is unsuitable for real-time conversation. Since relying solely on the current conversation state is insufficient to infer the long-term impact of routing decisions, motivated by the idea that approximated future states can provide guidance on potential conversation evolution and ease routing decisions, we introduce a retrieval-based future state approximation derived from search trajectories to assist routing. Through policy learning, ConvoRouter distills long-horizon knowledge from search, enabling efficient real-time routing.

Our main contributions are summarized as follows: (i) To the best of our knowledge, we are the first to investigate long-horizon routing in multi-turn conversation settings with continuously evolving user queries. (ii) We propose ConvoRouter, which leverages MCTS-discovered trajectories and behavior cloning with retrieval-based future state approximation to enable high-quality, real-time routing for multi-turn conversation. (iii) We extend the reward formulation to incorporate LLM inference costs and KV cache reuse, facilitating performance-cost trade-offs. (iv) Extensive experiments across three conversation domains and multiple open- and closed-source LLM candidate sets show that ConvoRouter consistently outperforms the strongest single LLMs and existing routing baselines in task success rate, while achieving superior performance-cost trade-offs under cost-aware settings.

2 Related Work

LLM Routing. Existing LLM routing methods mainly target single-turn interaction. Representative works such as RouteLLM [29] and EmbedLLM [44] learn compact query and model representations via matrix factorization (MF) to predict routing correctness. IRT-Router [34] models queries using item response theory, with two variants, MIRT and NIRT. RouterDC [4] applies dual contrastive learning to capture query–model compatibility. RouterBench [17] systematically evaluates a range of simple yet effective routing strategies, including KNN-based methods. Avengers [42] proposes a training-free routing approach that performs model selection through query embedding clustering and response score ranking. Router-R1 [40] decomposes a single fixed query into sub-queries and performs routing over these sub-queries. GMTRouter [36] leverages user histories to enable personalized routing decisions. However, none of them explicitly addresses routing in multi-turn conversations with continuously evolving user queries. Although Blending [26] considers multi-turn dialogue with multiple LLMs, it adopts random response selection instead of explicitly learning routing decisions. In contrast, ConvoRouter adopts a holistic multi-turn conversation perspective and formulates LLM routing as a long-horizon decision-making problem, explicitly modeling the impact of LLM selection on future conversation evolution and final user intent fulfillment.

Multi-turn Dialogue Planning. Different from LLM routing, which selects among multiple LLMs, existing work on multi-turn dialogue planning typically focuses on a single LLM and explores search or learning over predefined prompting strategies to solve cooperative or adversarial dialogue tasks. Representative methods include GDP-Zero [39], which combines MCTS with LLM prompting; PPDPP [11], which trains an external planner via reinforcement learning; and approaches inspired by AlphaZero [32], such as DPDP [16] and DMNA [24], which integrate MCTS with policy learning for dialogue planning. However, these methods operate within a fixed action space of prompting strategies rather than selecting among heterogeneous LLMs. In contrast, our work studies LLM routing in multi-turn conversations, where model selection becomes a key decision in planning.

3 Preliminaries

3.1 Multi-Turn Conversation and User Intent Fulfillment

A multi-turn conversation can be represented as an interaction sequence $\tau_T = [x_1, y_1, x_2, y_2, \ldots, x_T, y_T]$, where x_t and y_t denote the user input and the system response at turn t, respectively, and T is the terminal turn of the conversation.

To characterize the conversation-level fulfillment of user intents in τ_T, inspired by fine-grained evaluation criteria [9, 35], we explicitly decompose the user's high-level goal into a set of assessable task requirements, forming a conversation-level checklist $C = [c_1, c_2, \ldots, c_n]$, where each item c_i corresponds to a key constraint or sub-goal associated with the user's intent over the course of the conversation. At the end of the conversation, each checklist item is evaluated based on the overall conversation trajectory as *fulfilled*, *partially fulfilled*, or *unfulfilled*, and is assigned a score $r_i^{(T)} \in \{1, 0.5, 0\}$, respectively, yielding the checklist completion score $R_T = \sum_{i=1}^{n} r_i^{(T)}$. The *partially fulfilled* signal captures incremental intent progress that would be missed by a binary reward. Checklist construction details are provided in Appendix C.4.

3.2 LLM Routing Problem Formulation in Multi-Turn Conversation

LLM routing in multi-turn conversations is formulated as a sequential decision-making problem, where at each turn the router selects an LLM from a candidate set $\mathcal{M} = \{M_1, \ldots, M_N\}$ to generate a system response with the objective of maximizing the conversation-level completion score R_T. At turn t, the router observes the current state $s_t = [\tau_{t-1}, x_t]$, consisting of the conversation history up to the previous turn τ_{t-1} and the current user input x_t. Given the state s_t, the router selects a LLM from $\mathcal{M}$ to produce a system response. We denote this selection as the action a_t, and the response is generated as $y_t = M_{a_t}(s_t)$, after which the conversation history is updated to $\tau_t = [\tau_{t-1}, x_t, y_t]$. The user subsequently issues the next query x_{t+1} based on the current conversation τ_t and the router observes the updated state $s_{t+1} = [\tau_t, x_{t+1}]$, entering the next turn of LLM selection.

Since checklist items may be progressively addressed over the conversation, we define R_t as the checklist completion score at τ_t (with $R_0 = 0$), and define the immediate reward for action a_t in state

s_t as the incremental checklist completion,

$$\mathcal{R}(s_t, a_t) = R_t - R_{t-1}. \tag{1}$$

Under this formulation, the objective of multi-turn LLM routing is to learn a routing policy π_R that maximizes the expected cumulative reward,

$$\max_{\pi_R} \mathbb{E}\left[\sum_{t=1}^{T} \mathcal{R}(s_t, a_t)\right]. \tag{2}$$

This objective is equivalent to maximizing the conversation-level completion score R_T, since R_T admits an additive decomposition across turns as $R_T = \sum_{t=1}^{T}(R_t - R_{t-1})$.

In contrast, single-turn routing focuses on selecting, at each interaction, the LLM that maximizes the immediate reward, without accounting for the impact of LLM choices on future interactions. In multi-turn conversations, however, LLM response y_t influences subsequent user behavior x_{t+1}, rendering routing strategies based only on immediate rewards inherently myopic and often suboptimal with respect to conversation-level objectives. Only in the special case where future user inputs are independent of the preceding conversation context does multi-turn conversation LLM routing degenerate into a set of independent single-turn routing problems.

4 ConvoRouter

In this section, we present **ConvoRouter**, a LLM routing framework for multi-turn conversation, as illustrated in Figure 2. Section 4.1 describes search-derived trajectory generation, where MCTS with a user simulator is used to explore conversation branches induced by different LLM selections and collect routing decision trajectories with high cumulative rewards. Section 4.2 describes routing policy learning, where a routing model is trained via behavior cloning on the trajectories and augmented with a retrieval-based future state approximation, enabling routing without search at inference time.

4.1 Simulation-Based Search Trajectory Generation

User Query Simulation. In multi-turn conversation settings, routing decisions exhibit strong dynamics, as responses generated by different LLMs at the same turn can induce different user behaviors in subsequent turns, leading the conversation to evolve along different trajectories and affecting the overall conversation outcome. To capture this dynamic interaction process, we adopt an LLM-based user simulator to generate subsequent user responses. Specifically, the user simulator is responsible for producing the next user query and determining conversation termination; conditioned on a predefined user profile p_{usr} and the current conversation history τ_t, it generates the next user input as $x_{t+1} = M_{\text{usr}}(p_{\text{usr}}, \tau_t)$.

LLM Response Branch Exploration. Given a mechanism for simulating subsequent user queries, the remaining challenge is to systematically explore the conversation branches induced by different LLM responses and assess their long-term impact on conversation outcomes. We adopt MCTS to search over routing decision sequences. In the search process, each conversation state s_t is treated as a node in the search tree, with the current state s_{root} as the root. For a given state, candidate LLMs correspond to different action branches, which are expanded by invoking the selected model to generate a system response. Each generated response is evaluated by an LLM-based reward model: given the task checklist C, the current conversation state s_t, and the system response y_t, the reward model outputs per-item completion scores $[r_1^{(t)}, \ldots, r_n^{(t)}] = M_{\text{rew}}(C, s_t, y_t)$, which form the basis for computing path-level returns. A user simulator then generates the next user input, driving the conversation state to the next turn. By repeatedly expanding branches in the simulated environment, evaluating turn-level completions with the reward model, and advancing conversation evolution via the user simulator, MCTS compares the cumulative returns of conversation paths and identifies routing decision paths with superior overall performance. For each conversation task in the training set, MCTS yields a routing decision sequence, which is decomposed into state–action pairs to construct a search-derived dataset $\mathcal{D}_{\text{search}} = \{(s_t, a_t^*)\}$, where a_t^* denotes the routing action selected by MCTS at state s_t.

Details of the user simulator, reward model, and MCTS are provided in Appendices L and B.

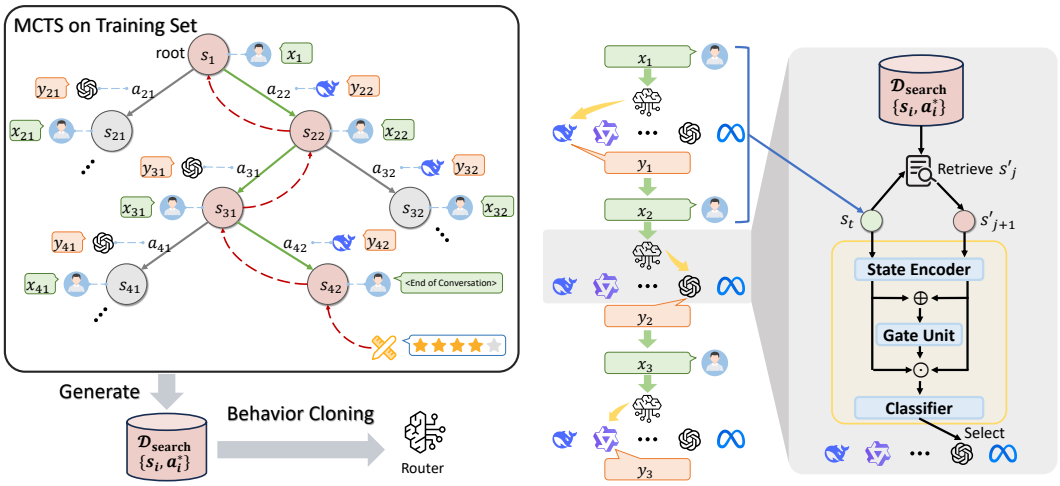


Figure 2: **Left**: Overview of ConvoRouter. MCTS with a user simulator and reward model is used to explore conversation branches and collect high-quality routing trajectories, which are used for offline policy learning via behavior cloning to train a search-free router. **Right**: Routing process of ConvoRouter in multi-turn conversation. At each turn, current conversation state s_t and a retrieved future state approximation s'_{j+1} from $\mathcal{D}_{\text{search}}$ are jointly encoded and fused to predict LLM selection.

4.2 Search-Guided Routing Policy Learning

Retrieval-based Future State Approximation. The routing model $\pi_R(a|s)$ takes the current conversation state as input and outputs a predicted probability distribution over actions. However, in the search-derived data, the optimal action a^* is obtained through search and corresponds to the conversation branch that maximizes future cumulative rewards, making it challenging for the routing model to directly learn such long-horizon dependencies. Inspired by prior work [31, 15], we posit that if the router could "imagine" an ideal future conversation state when making the current decision, even in the form of a coarse-grained semantic representation, the difficulty of routing would be substantially reduced. To this end, we introduce a long-term-aware state modeling strategy that augments the current state s_t with an approximate future state $\tilde{s}_{t+1}$, capturing the potential trend of future conversation evolution.

However, the real next state s_{t+1} is unobservable, and predicting future states via parameterized generation in natural language or semantic space is difficult for a lightweight routing model. Therefore, we adopt a retrieval-based approximation. For a given current state s_t, we employ a fixed retriever Ret to retrieve a semantically similar state $s'_j = \mathrm{Ret}(\mathcal{D}_{\text{search}}, s_t)$ from the search-derived dataset $\mathcal{D}_{\text{search}}$, and use its successor state s'_{j+1} along the corresponding search-derived trajectory as an approximation of the future evolution of s_t. This approximate future state is incorporated as an auxiliary signal during training and is likewise used at inference time, enabling the routing policy to capture potential future conversation trends under limited current semantic information.

Routing Model Architecture. As shown in Figure 2 (Right), we use a conversation state encoder $E(\cdot)$ to encode the current state and the retrieved approximate future state as

$$e_t = E(s_t), \quad \tilde{e}_{t+1} = E(s'_{j+1}). \tag{3}$$

To balance the information from the current state and the potentially noisy retrieved future state, we introduce a trainable gating mechanism for adaptive fusion:

$$g = \sigma\big(W_G[e_t, \tilde{e}_{t+1}]\big), \quad h = g \odot e_t + (1 - g) \odot \tilde{e}_{t+1}. \tag{4}$$

The fused representation h is then fed into the classification head $\mathrm{Cls}(\cdot)$ to produce the final routing policy output $\pi_R(\cdot|s_t)$, based on which the candidate model is selected.

Long-term-aware Supervision. As the training dataset $\mathcal{D}_{\text{search}}$ is obtained via planning with respect to long-term cumulative rewards, it provides supervision signals that are aligned with long-term outcomes, enabling long-term-aware learning of the routing policy. We adopt a behavior cloning approach, treating the MCTS routing policy π_{MCTS} as the expert and training the routing model π_R to imitate its decision behavior, as it is simple and efficient and suits the real-time requirements of

conversations. The training objective is to minimize the cross-entropy loss between π_R and the π_{MCTS} actions:

$$\min_{\theta} -\frac{1}{|\mathcal{D}_{\text{search}}|} \sum_{(s,a)\in\mathcal{D}_{\text{search}}} \log \pi_R(a \mid s;\theta). \tag{5}$$

In this way, the router can absorb the long-horizon planning knowledge implicit in MCTS during training, and perform long-term decision at inference time without explicit search.

5 Extending Reward Function with Cost

LLM routing decisions jointly impact conversation quality and monetary cost, motivating the incorporation of cost modeling into the reward function. To this end, we extend the original reward design with an explicit cost component. Rather than treating cost as a simple linear function, our cost formulation is directly grounded in the publicly available API pricing rules[3] of major cloud providers.

Specifically, closed-source LLMs typically employ a KV cache mechanism to store historical context: when the same model is invoked in consecutive turns, cached context tokens are charged at a lower cost, whereas switching models invalidates the cache[4], requiring the entire conversation history to be re-encoded and billed at the standard input cost. As a result, the cost in multi-turn conversation routing depends not only on which model is selected, but also on the sequence of model choices.

For each candidate model M_i, the response cost consists of two components: output and input cost. Let $c_{\text{out}}^{(i)}, c_{\text{in}}^{(i)}, c_{\text{cache}}^{(i)}$ denote the per-token output cost, standard input cost, and cache-hit input cost of M_i, respectively, with $c_{\text{cache}}^{(i)} < c_{\text{in}}^{(i)}$. Suppose at conversation turn t, the previous conversation history τ_{t-1}, the current user query x_t, and the response y_t generated by model M_{a_t} have token counts $L(\tau_{t-1})$, $L(x_t)$, and $L(y_t)$, respectively. The output cost reward is defined as: $R_t^{\text{out}} = -c_{\text{out}}^{(a_t)} \cdot L(y_t)$.

The input cost accounts for cache hits: if the same model is used in consecutive turns, cached history tokens is charged at the lower cache cost, while the remaining tokens are charged at the standard input cost; if the model switches, all tokens are charged at the standard input cost:

$$R_t^{\text{in}} = \begin{cases} -\Big[c_{\text{in}}^{(a_t)}\big((L(\tau_{t-1}) \bmod B) + L(x_t)\big) + c_{\text{cache}}^{(a_t)}\Big\lfloor \frac{L(\tau_{t-1})}{B} \Big\rfloor \cdot B\Big], & a_t = a_{t-1} \\ -c_{\text{in}}^{(a_t)}\big(L(\tau_{t-1}) + L(x_t)\big), & a_t \neq a_{t-1} \end{cases} \tag{6}$$

where B is the token size of a KV cache block. The total cost reward is then defined as $R_t^{\text{cost}} = R_t^{\text{in}} + R_t^{\text{out}}$. Accordingly, the original reward function (Eq. 1) can be extended as:

$$\mathcal{R}(s_t, a_t) = R_t - R_{t-1} + \lambda \cdot R_t^{\text{cost}}, \tag{7}$$

where λ is a hyperparameter controlling the trade-off between conversation quality and cost. This extension allows MCTS search and routing policy training to jointly consider performance and cost.

6 Experiments

6.1 Experimental Setup

Datasets. We conduct experiments on multi-turn conversation tasks using three real-world datasets, including one open-domain dataset, **ShareGPT**[5], collected from real user–LLM conversations, and two domain-specific datasets, **JDDC** (Jing Dong Dialogue Corpus) [3], collected from real-world e-commerce customer-service interactions, and **MedDG** [23], collected from real online medical consultation scenarios. We extract user profiles and checklists from each conversation instance to construct user simulators and reward model, enabling interactive environments for corresponding multi-turn conversation tasks, following the setup of [11, 16] and aligning with their data scale.

[3] `https://openai.com/index/api-prompt-caching`

[4] Due to the complexity of real-world conversations, we consider the worst-case scenario for model switching, in which the KV cache is assumed to be completely invalidated. We further investigate partial cache sharing across different LLMs and imperfect cache reuse under consecutive calls to the same LLM in Appendices I.3 and I.4, respectively.

[5] `https://sharegpt.com`

Table 1: Performance comparison of ConvoRouter and various baselines across different conversation tasks using Qwen and Llama candidate sets. Results of stochastic methods (Random, Greedy Router, and MCTS Router) are averaged over three random seeds. The best results among practical methods are shown in **bold**, and the second-best are underlined.

Method	Qwen Series Set							Llama Series Set						
	ShareGPT		JDDC		MedDG		SR_{avg} ↑	ShareGPT		JDDC		MedDG		SR_{avg} ↑
	SR↑	AT↓	SR↑	AT↓	SR↑	AT↓		SR↑	AT↓	SR↑	AT↓	SR↑	AT↓	
Qwen3-4B / Llama3.2-1B	81.75	5.09	73.07	3.83	70.33	4.98	75.05	63.03	6.62	48.16	4.13	42.13	6.84	51.11
Qwen3-8B / Llama3.2-3B	82.80	5.03	75.23	3.70	71.14	5.08	76.39	73.77	5.92	71.55	3.75	62.42	6.23	69.25
Qwen3-14B / Llama3.1-8B	83.24	5.11	76.41	3.48	71.38	5.18	77.01	79.24	**5.60**	72.92	3.61	64.93	**6.01**	72.36
Random [26]	81.50	4.95	76.82	**3.47**	71.04	4.91	76.45	76.16	6.12	69.63	4.17	61.58	6.29	69.12
Prompt LLM [13]	82.85	**4.92**	75.77	3.63	71.97	5.34	76.86	79.19	5.79	72.43	4.02	64.98	6.25	72.20
KNN Router [17]	83.37	5.01	77.78	3.48	70.95	**4.71**	77.37	81.71	5.77	74.89	3.56	65.36	6.13	73.99
MF Router [29]	82.41	5.13	76.39	3.75	71.60	4.77	76.80	80.01	5.67	71.88	3.67	66.27	6.42	72.72
MIRT [34]	83.30	5.05	76.43	3.81	71.08	5.02	76.94	80.17	5.65	73.38	3.75	66.48	6.51	73.34
NIRT [34]	82.61	5.11	76.64	3.65	70.56	5.17	76.60	80.56	5.78	72.94	3.64	65.73	6.85	73.08
RouterDC [4]	82.72	4.94	76.41	3.48	71.38	5.18	76.84	80.94	5.68	74.01	**3.52**	65.74	6.59	73.56
Avengers [42]	83.93	5.00	77.89	3.66	70.98	5.21	77.60	80.38	5.61	74.63	3.75	65.99	6.78	73.67
ConvoRouter (Ours)	**87.46**	4.96	**83.06**	3.52	**78.42**	4.86	**82.98**	**85.82**	5.72	**80.86**	3.82	**71.43**	6.23	**79.37**
Greedy Router	88.50	5.12	83.20	3.56	78.63	5.00	83.44	85.75	5.73	81.90	3.70	73.21	6.28	80.29
MCTS Router	94.11	4.97	88.26	3.66	86.18	4.72	89.52	91.94	5.76	85.97	3.98	82.64	6.55	86.85

To evaluate domain generalization, we hold out a subset of ShareGPT centered on legal and financial topics, forming a domain-generalization test set, denoted as **ShareGPT-LF**, which is completely non-overlapping with the training data. Further details of the datasets are provided in Appendix C.

Candidate LLM Sets. We selected mainstream large language models of different series and scales, forming four candidate sets: (i) **Qwen series** [37]: Qwen3-4B, Qwen3-8B, Qwen3-14B; (ii) **Llama series** [14]: Llama3.2-1B-Instruct, Llama3.2-3B-Instruct, Llama3.1-8B-Instruct; (iii) **Mixed series**: a combination of six models from the Qwen and Llama series; (iv) **Closed-source LLMs**: Qwen3-Max, DeepSeek-Chat-V3.2, GPT-5.1. Implementation details and pricing information are provided in Appendix D; among the closed-source LLMs, DeepSeek is priced significantly lower than the others.

Evaluation Metrics. We use **Success Rate (SR)** to measure the completion of conversation tasks. For a single task, SR is defined as the average completion of checklist items at the final turn T: $SR = \frac{1}{n}\sum_{i=1}^{n} r_i^{(T)}$, reflecting the cumulative reward without considering costs, where a higher SR indicates better performance. **Average Turns (AT)** measures the mean number of conversation turns required to complete a task, with lower values being better. For the closed-source LLM, we further introduce **Cost**, the total expense of LLM invocations over a dataset based on API pricing rules.

Baselines. We compare ConvoRouter with a range of baseline methods, including: **Single Fixed LLM**, **Random LLM** [26], **Prompt-based Router** [13], **RouterDC** [4], **KNN Router** [17], **Matrix Factorization** [29, 44], **IRT-Router** [34] and **Avengers** [42]. We further include two simulation-based routing methods that utilize the user simulator and reward model: **Greedy Router**, which performs one-step greedy search, and **MCTS Router**, which conducts MCTS.

Implementation Details. For MCTS, we use $K = 10$ simulations, an exploration coefficient $c = 2$, and a discount factor $\gamma = 0.999$. The maximum conversation turn is set to $T_{\max} = 8$. For retrieval-based future state approximation, we employ BGE-M3 [2] as the fixed retriever. The ConvoRouter is trained with a Qwen2.5-0.5B-based state encoder, where all but the last three transformer layers are frozen to mitigate overfitting. The user simulator and the reward model are implemented using Qwen3-Max. The router is optimized using AdamW [25] with learning rate of $1e{-5}$, batch size of 32, and trained for 15 epochs. All experiments are conducted on a single NVIDIA RTX 4090D GPU.

6.2 Main Results

Oracle Analysis. The results in Table 1 reveal a clear tiered performance pattern, characterized by a large performance gap between the MCTS and the Greedy Router, as well as a consistent gap between the Greedy Router and the strongest single LLMs. Although simulation-based routing is impractical at inference time, the MCTS Router provides a search-based reference that serves as an approximate, oracle-like upper bound on achievable performance under the current candidate set. Notably, the consistent gap between MCTS and the Greedy Router further highlights the critical role of long-term awareness in multi-turn conversation routing.

Table 2: Performance comparison of ConvoRouter and baselines on ShareGPT using mixed candidate set of Qwen and Llama models.

Method	SR↑	AT↓
Random [26]	79.50	5.74
KNN Router [17]	84.23	5.22
Avengers [42]	84.72	**5.12**
ConvoRouter (Ours)	**88.32**	5.21
Greedy Router	90.06	5.17
MCTS Router	94.99	5.13

Table 3: Cross-domain generalization results on ShareGPT-LF.

Method	Qwen Series Set		Llama Series Set	
	SR↑	AT↓	SR↑	AT↓
Qwen3-4B / Llama3.2-1B	80.24	4.80	62.25	6.44
Qwen3-8B / Llama3.2-3B	80.66	4.68	75.17	5.36
Qwen3-14B / Llama3.1-8B	81.74	4.65	80.18	**5.09**
Random [26]	81.03	4.67	77.62	5.76
KNN Router [17]	81.64	**4.56**	81.47	5.15
Avengers [42]	81.30	4.67	80.18	**5.09**
ConvoRouter (Ours)	**83.69**	4.64	**83.03**	5.48
Greedy Router	87.18	4.53	86.47	5.26
MCTS Router	91.46	4.50	90.07	5.14

Comparison with Single LLMs. Table 1 reports the performance of ConvoRouter on multi-turn conversation tasks over the Qwen and Llama candidate sets. ConvoRouter achieves an average success rate of 82.98% on the Qwen series, surpassing Qwen3-14B by 5.97%, and 79.37% on the Llama series, exceeding Llama3.1-8B-Instruct by 7.01%, consistently outperforming the strongest single LLM in each set. These results indicate that ConvoRouter can effectively exploit the complementary strengths of heterogeneous models, thereby breaking the performance ceiling of individual LLMs.

Comparison with Baseline Routers. Compared with representative routing baselines, ConvoRouter improves the average success rate by more than 5% over KNN Router, Avengers, and RouterDC across both candidate sets, and achieves performance comparable to the simulation-based Greedy Router. This aligns with our motivation that long-term awareness is crucial for effective multi-turn conversation routing beyond myopic decisions, showing that learning a router grounded in long-horizon planning enables high-quality routing without requiring online search. Although the average turns slightly increases in some scenarios, ConvoRouter maintains consistently higher success rates.

Adaptability on Mixed Series Candidate Sets. To evaluate the generalization and decision robustness of ConvoRouter when the candidate set includes multiple LLM families, we conducted experiments on ShareGPT with the Mixed series, which combines the Qwen and Llama series. As shown in Table 2, ConvoRouter achieves an SR of 88.32%, surpassing the baseline routers by over 3.60%, demonstrating its strong adaptability in selecting among heterogeneous LLM architectures. Notably, ConvoRouter performs even better on the mixed candidate set than on single-series candidate sets, indicating its ability to cross LLM families and effectively identify and integrate the complementary strengths of different LLMs within a larger candidate pool. We further evaluate ConvoRouter on a larger heterogeneous candidate pool of nine LLMs, with results presented in Appendix F.2.

Cross-Domain Generalization. To evaluate the cross-domain generalization of ConvoRouter, we conduct experiments on ShareGPT-LF. As shown in Table 3, all methods are trained on ShareGPT, which excludes legal and financial topics. ConvoRouter achieves success rate improvements of 1.95%–3.45% on the Qwen series and 1.56%–2.85% on the Llama series over the best single LLM and other baselines. These results demonstrate that ConvoRouter maintains a performance advantage in unseen domains, indicating that the learned routing policy can effectively leverage LLM complementarity to achieve robust cross-domain generalization.

Cross-Dataset Generalization and Additional Baseline Comparisons. To further evaluate the generalization of ConvoRouter, we conduct cross-dataset experiments on 1,000 multi-turn conversations from UltraChat [12] using the Qwen candidate set. ConvoRouter is trained on ShareGPT and directly evaluated on UltraChat without retraining or adaptation, while the retrieval bank remains unchanged. In addition, we employ DeepSeek-V4-Flash as the user simulator and GPT-5.1 as the evaluator on UltraChat, differing from the original evaluation setting.

We further introduce two baselines. GPT Router, inspired by GPT-5's real-time routing mechanism [33], prompts GPT-5-mini to assess conversation progress and candidate LLM suitability, integrating dialogue history, previous model selections, and model profiles to make routing decisions. BPPO [45] is an offline reinforcement learning method that optimizes routing policies using MCTS-generated trajectories and advantage estimates.

Table 4: Task success rate comparison on ShareGPT and UltraChat.

Method	ShareGPT	UltraChat
Qwen3-14B	83.24	66.75
KNN Router [17]	83.37	68.60
Avengers [42]	83.93	68.87
GPT Router	83.84	67.43
BPPO Router [45]	84.11	68.36
ConvoRouter (Ours)	**87.46**	**70.08**

As shown in Table 4, ConvoRouter consistently outperforms the additional baselines on both ShareGPT and UltraChat. In particular, it achieves a success rate of 70.08% on UltraChat, outperforming all baselines by 1.21–3.33%. These results demonstrate the effectiveness of ConvoRouter in cross-dataset generalization and its advantage over the offline policy optimization baseline.

Retrieved States Provide Reliable Approximations and are Effectively Utilized by the Gating Mechanism. We measure the similarity between retrieved states and the actual future states. As shown in Table 5, the average similarity on ShareGPT is around 0.7, while domain-specific datasets exhibit higher similarity and lower variance, empirically supporting that retrieved states provide meaningful approximations. Moreover, even when retrieval is imperfect, the gating unit can selectively exploit retrieved features. As illustrated in Figure 3, the gate activation ratio is positively correlated with retrieval similarity (Pearson $r = 0.53$), indicating that the gating mechanism can effectively identify high-quality retrievals and dynamically adjust their contribution according to the reliability of the approximated features.

Table 5: Mean and variance of retrieval similarity across datasets.

	ShareGPT	JDDC	MedDG
Mean	0.6958	0.7898	0.8046
Variance	0.0187	0.0048	0.0024

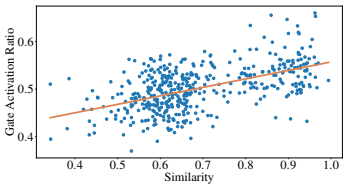


Figure 3: Correlation between retrieval similarity and gate activation.

6.3 Results of Cost-aware Routing

Table 6: Performance and cost comparison of ConvoRouter and baselines on ShareGPT using closed-source candidate set. Cost-aware rewards are applied in the Balance setting. Cost ($) refers to the total monetary cost for the test set.

Method	Performance First			Balance		
	SR↑	AT↓	Cost↓	SR↑	AT↓	Cost↓
Qwen3-Max	84.09	**4.53**	0.30	84.09	**4.53**	0.30
DeepSeek-Chat-V3.2	85.13	4.66	**0.04**	85.13	4.66	**0.04**
GPT-5.1	84.57	4.80	1.05	84.57	4.80	1.05
Random [26]	84.41	4.57	0.52	84.41	4.57	0.52
KNN Router [17]	85.91	4.56	0.50	85.13	4.66	**0.04**
Avengers [42]	84.54	4.56	0.32	85.13	4.66	**0.04**
ConvoRouter (Ours)	**89.92**	4.59	0.44	**88.69**	4.63	0.09
Greedy Router	91.80	4.66	0.53	91.28	4.43	0.20
MCTS Router	96.11	4.52	0.46	95.93	4.44	0.19

Table 6 reports the results of ConvoRouter on the closed-source LLM candidate set with cost reward on ShareGPT. Under the Performance First setting ($\lambda = 0$), ConvoRouter achieves the highest success rate of 89.92%, substantially outperforming strong single closed-source LLMs such as DeepSeek-Chat-V3.2 (+4.97%) and GPT-5.1 (+5.35%), as well as all routing baselines. Under the Balance setting ($\lambda = 50$), incorporating cost reward reduces the average cost from 0.44 to 0.09, corresponding to an approximately 80% reduction, while the success rate only decreases to 88.69%, amounting to a relative drop of about 1.4%. In contrast, several baselines exhibit clear policy degradation, effectively collapsing to a single low-cost DeepSeek selection, thereby abandoning the decision-making role expected of a router. These results demonstrate that, with cost reward, ConvoRouter achieves a more balanced performance-cost trade-off than baseline methods, substantially reducing inference cost with only a limited degradation in task performance. Analysis of weight λ is provided in Appendix I.2.

6.4 Ablation Study

Effect of MCTS. As shown in Table 7, removing MCTS (w/o MCTS) and training the router solely with one-step greedy decisions results in an average SR drop of 3.71%. This shows that relying on myopic routing decisions is insufficient to capture long-term effects in multi-turn conversations, whereas MCTS-generated data provide long-horizon supervision critical to ConvoRouter's performance gains.

Effect of Retrieval-based Future State Approximation. Table 7 also shows that removing the retrieval module (w/o Ret) or using random retrieval (Random Ret) reduces the average SR by 3.49%

and 4.37%, respectively, both significantly lower than the complete model. The larger performance drop with random retrieval is primarily due to the introduction of irrelevant information, which amplifies noise and further impairs routing decisions. In contrast, semantically relevant retrieved future states provide more targeted cues, helping the router capture potential conversation evolution trends and mitigating the difficulty of making long-term decisions based solely on the current state, while incurring only a negligible retrieval overhead of 0.01s per decision.

Table 7: Ablation study of ConvoRouter components on success rate, showing the effects of removing MCTS-based policy discovery, the retrieval module, and using alternative information fusion strategies across three conversation datasets using the Qwen candidate set.

Method	ShareGPT	JDDC	MedDG	SR_{avg} ↑
ConvoRouter (Ours)	87.46	83.06	78.42	82.98
w/o MCTS	83.64	80.21	73.97	79.27
w/o Ret	84.78	79.66	74.03	79.49
Random Ret	83.99	79.16	72.67	78.61
ConvoRouter (Add Fus.)	85.27	80.62	75.14	80.34
ConvoRouter (Concat Fus.)	84.85	80.75	75.58	80.39

Effect of Fusion Strategies for Future States. As shown in Table 7, incorporating future states without differentiation (Concat Fus. or Add Fus.) yields some improvement but underperforms the full model with gating. This suggests that state information from different sources varies in reliability and relevance, and a learnable gating mechanism to adaptively fuse them helps suppress retrieval noise and balance current and future approximated information, resulting in more robust routing.

7 Conclusion

In this paper, we study LLM routing in multi-turn conversation settings and propose ConvoRouter, a long-horizon routing approach for multi-turn conversation. ConvoRouter learns routing policies from MCTS-based data, enabling long-horizon routing decisions without online search, and further incorporates retrieval-based approximate future states to assist LLM selection. Experiments across diverse multi-turn conversation tasks demonstrate that ConvoRouter effectively integrates the complementary strengths of multiple LLMs, significantly outperforming the strongest single LLMs and existing routing baselines based on local rewards in terms of task success rate; moreover, when augmented with cost-aware rewards, ConvoRouter achieves a more favorable performance-cost trade-off.

8 Limitations and Broader Impact

Limitations. Our evaluation is limited by the scale of available conversational data, particularly in open-domain settings, where identifying conversations with explicit goals and verifiable completion criteria remains challenging. Although our evaluation extends to a larger cross-dataset setting, further expanding the evaluation scale and dataset diversity remains a valuable direction for future work.

Broader Impact. From a broader perspective, we intend to open new avenues for future research and practice in LLM routing, encouraging both research and industrial communities to shift focus from isolated single-turn question answering to multi-turn conversational settings. Since multi-turn conversation represents more widely used scenarios, it naturally emerges as the core workload for LLM service providers. Routing will evolve from a stateless per-query selection into a dynamic orchestration of LLMs across the entire conversation session to fulfill user intents. Therefore, exploring effective LLM routing strategies for multi-turn conversation holds high practical value.

Acknowledgments and Disclosure of Funding

This work was supported in part by Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China JYB2025XDXM103, CCF-Tencent Rhino-Bird Open Research Fund (No. RAGR20260126), and SJTU-Huawei Explore X Gift Fund.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 2318–2335, 2024.

[3] Meng Chen, Ruixue Liu, Lei Shen, Shaozu Yuan, Jingyan Zhou, Youzheng Wu, Xiaodong He, and Bowen Zhou. The jddc corpus: A large-scale multi-turn chinese dialogue dataset for e-commerce customer service. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 459–466, 2020.

[4] Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems*, 37:66305–66328, 2024.

[5] Yirong Chen, Xiaofen Xing, Jingkai Lin, Huimin Zheng, Zhenyu Wang, Qi Liu, and Xiangmin Xu. Soulchat: Improving llms' empathy, listening, and comfort abilities through fine-tuning with multi-turn empathy conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1170–1183, 2023.

[6] Zhijun Chen, Jingzheng Li, Pengpeng Chen, Zhuoran Li, Kai Sun, Yuankai Luo, Qianren Mao, Ming Li, Likang Xiao, Dingqi Yang, et al. Harnessing multiple large language models: A survey on llm ensemble. *arXiv preprint arXiv:2502.18036*, 2025.

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[8] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[9] Jonathan Cook, Tim Rocktäschel, Jakob Foerster, Dennis Aumiller, and Alex Wang. Ticking all the boxes: Generated checklists improve llm evaluation and generation. *arXiv preprint arXiv:2410.03608*, 2024.

[10] Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pages 72–83. Springer, 2006.

[11] Yang Deng, Wenxuan Zhang, Wai Lam, See-Kiong Ng, and Tat-Seng Chua. Plug-and-play policy planner for large language model powered dialogue agents. In *ICLR*, 2024.

[12] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, 2023.

[13] Tao Feng, Yanzhen Shen, and Jiaxuan You. Graphrouter: A graph-based router for LLM selections. In *The Thirteenth International Conference on Learning Representations*, 2025.

[14] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[15] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[16] Tao He, Lizi Liao, Yixin Cao, Yuanxing Liu, Ming Liu, Zerui Chen, and Bing Qin. Planning like human: A dual-process framework for dialogue planning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4768–4791, 2024.

[17] Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. Routerbench: A benchmark for multi-llm routing system. *arXiv preprint arXiv:2403.12031*, 2024.

[18] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626, 2023.

[19] Yangning Li, Shirong Ma, Xiaobin Wang, Shen Huang, Chengyue Jiang, Hai-Tao Zheng, Pengjun Xie, Fei Huang, and Yong Jiang. Ecomgpt: Instruction-tuning large language models with chain-of-task tasks for e-commerce. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18582–18590, 2024.

[20] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022.

[21] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[22] Jiayu Liu, Zhenya Huang, Tong Xiao, Jing Sha, Jinze Wu, Qi Liu, Shijin Wang, and Enhong Chen. Socraticlm: Exploring socratic personalized teaching with large language models. *Advances in Neural Information Processing Systems*, 37:85693–85721, 2024.

[23] Wenge Liu, Jianheng Tang, Yi Cheng, Wenjie Li, Yefeng Zheng, and Xiaodan Liang. Meddg: an entity-centric medical consultation dataset for entity-aware medical dialogue generation. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 447–459. Springer, 2022.

[24] Yutong Liu, Lida Shi, Rui Song, and Hao Xu. A dual-mind framework for strategic and expressive negotiation agent. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23840–23860, 2025.

[25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[26] Xiaoding Lu, Zongyi Liu, Adian Liusie, Vyas Raina, Vineet Mudupalli, Yuwen Zhang, and William Beauchamp. Blending is all you need: Cheaper, better alternative to trillion-parameters llm. *arXiv preprint arXiv:2401.02994*, 2024.

[27] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 9802–9822, 2023.

[28] Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. Using an llm to help with code understanding. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13, 2024.

[29] Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025.

[30] Sungjin Park, Xiao Liu, Yeyun Gong, and Edward Choi. Ensembling large language models with process reward-guided tree search for better complex reasoning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 10256–10277, 2025.

[31] Razvan Pascanu, Yujia Li, Oriol Vinyals, Nicolas Heess, Lars Buesing, Sebastien Racanière, David Reichert, Théophane Weber, Daan Wierstra, and Peter Battaglia. Learning model-based planning from scratch. *arXiv preprint arXiv:1707.06170*, 2017.

[32] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[33] Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

[34] Wei Song, Zhenya Huang, Cheng Cheng, Weibo Gao, Bihan Xu, GuanHao Zhao, Fei Wang, and Runze Wu. Irt-router: Effective and interpretable multi-llm routing via item response theory. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15629–15644, 2025.

[35] Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. *arXiv preprint arXiv:2507.18624*, 2025.

[36] Encheng Xie, Yihang Sun, Tao Feng, and Jiaxuan You. Gmtrouter: Personalized llm router over multi-turn user interactions. *arXiv preprint arXiv:2511.08590*, 2025.

[37] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[38] Zihao Yi, Jiarui Ouyang, Zhe Xu, Yuwen Liu, Tianhao Liao, Haohao Luo, and Ying Shen. A survey on recent advances in llm-based multi-turn dialogue systems. *ACM Computing Surveys*, 58(6):1–38, 2025.

[39] Xiao Yu, Maximillian Chen, and Zhou Yu. Prompt-based monte-carlo tree search for goal-oriented dialogue policy planning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7101–7125, 2023.

[40] Haozhen Zhang, Tao Feng, and Jiaxuan You. Router-r1: Teaching llms multi-round routing and aggregation via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[41] Hongbo Zhang, Junying Chen, Feng Jiang, Fei Yu, Zhihong Chen, Guiming Chen, Jianquan Li, Xiangbo Wu, Zhang Zhiyi, Qingying Xiao, et al. Huatuogpt, towards taming language model to be a doctor. In *Findings of the association for computational linguistics: EMNLP 2023*, pages 10859–10885, 2023.

[42] Yiqun Zhang, Hao Li, Chenxu Wang, Linyao Chen, Qiaosheng Zhang, Peng Ye, Shi Feng, Daling Wang, Zhen Wang, Xinrun Wang, et al. The avengers: A simple recipe for uniting smaller language models to challenge proprietary giants. *arXiv preprint arXiv:2505.19797*, 2025.

[43] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

[44] Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran. EmbedLLM: Learning compact representations of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[45] Zifeng Zhuang, Kun LEI, Jinxin Liu, Donglin Wang, and Yilang Guo. Behavior proximal policy optimization. In *The Eleventh International Conference on Learning Representations*, 2023.

Appendix Contents

A Key Differences from Prior Work **15**

B Details of MCTS **15**

C Details of Datasets **16**

C.1 Dataset Sources and Characteristics 16

C.2 Data Splits 16

C.3 Topic Distribution of ShareGPT 17

C.4 User Profiling and Checklist Construction 17

C.5 Quality and Reliability Validation of LLM-Generated Checklists and User Profiles 17

D Details of Candidate LLMs **20**

D.1 Implementation of Candidate LLMs 20

D.2 Pricing of Closed-Source LLMs 20

D.3 Considerations on LLM Candidate Set Selection 21

E Training and Inference Time and GPU Resources of ConvoRouter **21**

F Additional Experiments on Candidate LLM Pools **21**

F.1 Effectiveness of Candidate Set Composition 21

F.2 Evaluation on a Larger Heterogeneous Candidate Pool 22

G Robustness to Potential Bias from LLM Choices in Evaluation and MCTS Guidance **22**

G.1 Consistency Across Different LLMs Used as Final Evaluators 22

G.2 Consistency Across Different LLMs Used for MCTS Guidance 22

H Human Evaluation **23**

I Details of Cost-Aware Routing **23**

I.1 Statistical Significance and Cost Analysis 23

I.2 Cost and Performance under Varying Cost Weights λ 23

I.3 Cost Formulation under KV Cache Sharing 24

I.4 Cost Formulation under Imperfect KV Cache Reuse 24

J Analysis of ConvoRouter's Routing Behavior **25**

K Case Studies **26**

L Prompting Details **28**

A Key Differences from Prior Work

Table 8: Comparison between ConvoRouter and prior work.

	Our Work (ConvoRouter)	Prior Work
Application Scenario	Multi-turn conversation	Single-turn interaction
Routing Granularity	A dynamically evolving sequence of user queries that are unknown ahead of conversation	A static single query or its internal reasoning steps
Optimization Objective	Global cumulative reward, i.e., the overall fulfillment of user intents across the entire conversation	Immediate reward, i.e., the correctness of a single query response
Decision Horizon	Long-horizon awareness enabled by MCTS-explored trajectories and retrieval-based future state approximation, considering the impact of current LLM selections on future conversation evolution and final outcomes	Myopic selection, optimizing only for the current turn while ignoring impacts on subsequent interactions
Real-Time Decision or Not	Search-free inference enabled by distillation from offline MCTS, suitable for real-time multi-turn conversation	Single-turn QA routing is real-time; step-level and online search methods incur high response latency
Cost Modeling	Dynamic cost modeling, explicitly accounting for KV cache invalidation and reuse caused by LLM switching	Static single-invocation cost calculation, without considering context caching
Datasets	Open-domain (ShareGPT), customer service (JDDC [3]), and medical consultation (MedDG [23]) conversations	Mathematical reasoning (GSM8K [7]), commonsense QA (PopQA [27]), etc.

Table 8 summarizes the key differences between our work and prior studies. We focus on multi-turn conversation, the most practical and natural interaction paradigm between users and LLMs. In this setting, user queries continuously evolve and are interdependent across turns, implying that routing decisions at the current turn can influence future user queries and thus the overall outcome. This introduces both dynamic interaction effects and delayed rewards, making per-turn optimal selection inconsistent with globally optimal decisions. These characteristics motivate our long-horizon routing framework that explicitly optimizes cumulative conversation reward. In contrast, existing routing methods mainly target single-turn interaction or static queries with globally known inputs, such as step-level reasoning or tool-use scenarios, limiting their applicability to multi-turn conversation.

In multi-turn conversations, routing additionally introduces requirements for real-time decision making and modeling of switching-induced cost dynamics across LLM invocations. To meet the real-time requirement, our method distills policies from offline MCTS, enabling lightweight inference for real-time conversation, with 0.03 seconds per query. In contrast, prior work based on step-level routing and online search exhibits significantly higher inference latency. LE-MCTS [30] relies on online search, resulting in approximately 100 seconds per query, while Router-R1 [40] incurs additional overhead due to multiple LLM invocations per query. These methods are therefore unsuitable for real-time conversation scenarios. For cost modeling, we explicitly account for KV cache invalidation and reuse induced by LLM switching, whereas existing methods rely on simplified linear cost assumptions that fail to capture caching effects in multi-turn conversation.

B Details of MCTS

In this section, we describe the details of Monte Carlo Tree Search (MCTS) [10, 32]. Starting from the current conversation state s_{root} as the root node, MCTS repeatedly performs four phases—Selection, Expansion, Simulation, and Backpropagation—for a total of K iterations, progressively exploring conversation trajectories induced by different model selections and estimating their long-term returns.

Selection. Starting from the root node s_{root}, MCTS recursively selects child nodes until reaching a leaf node that is not fully expanded. Since no reliable policy prior is available during the search stage, we adopt the standard Upper Confidence Tree (UCT) [32] criterion, which provides an unbiased balance between exploration and exploitation:

$$UCT(s,a) = Q(s,a) + c\sqrt{\frac{\ln N(s)}{N(s,a)+1}}, \tag{8}$$

where $Q(s, a)$ denotes the estimated cumulative return of taking action a at state s, $N(s)$ is the number of visits to state s, $N(s, a)$ is the number of times action a has been selected at state s, and c is a constant controlling the degree of exploration. At each step, the action maximizing $UCT(s, a)$ is selected.

Expansion. Upon reaching a leaf node, MCTS selects an unexplored action, corresponding to a model routing decision for the next conversation turn, and generates a new conversation state, which is then added to the search tree.

Simulation. From the newly expanded node, MCTS performs an approximate rollout to estimate its potential return. Specifically, we adopt a one-step greedy routing strategy, selecting at each turn the currently estimated optimal model, and advance the conversation using the user simulator until termination (e.g., when the user simulator outputs a termination signal or the maximum conversation length $T_{\max}$ is reached).

Backpropagation. After a simulation terminates, the accumulated return is propagated backward along the simulated trajectory from the leaf node to the root. For each state s_t on the trajectory, the return is defined as the discounted cumulative reward from that state until the end of the conversation:

$$V(s_t) = \sum_{k=t}^{T} \gamma^{k-t} \mathcal{R}(s_k, a_k), \tag{9}$$

where $\gamma \in (0, 1]$ is the discount factor. Subsequently, for each encountered state–action pair (s, a), the visit count and value estimate are updated as:

$$\begin{aligned} N(s, a) &\leftarrow N(s, a) + 1, \\ Q(s, a) &\leftarrow Q(s, a) + \frac{V(s) - Q(s, a)}{N(s, a)}. \end{aligned} \tag{10}$$

After completing all K search iterations, we select, at the root node s_{root}, the action with the highest estimated value as the optimal model choice for the current state:

$$a^* = \arg\max_a Q(s_{\text{root}}, a). \tag{11}$$

C Details of Datasets

C.1 Dataset Sources and Characteristics

ShareGPT[6] is an English open-domain multi-turn conversation dataset collected from real user–LLM interactions, where users share conversation links generated on the ChatGPT web interface. The dataset covers a wide range of general-purpose tasks, including writing assistance, code generation, and daily conversations, reflecting diverse real-world usage scenarios of conversational AI systems.

JDDC (Jing Dong Dialogue Corpus) [3] is a Chinese e-commerce customer-service multi-turn dialogue dataset released by JD.com, derived from real interaction logs between users and customer-service systems. The dataset focuses on task-oriented dialogue scenarios in the e-commerce domain, covering business processes such as product inquiry, order management, and after-sales support.

MedDG [23] is a Chinese medical multi-turn conversation dataset collected from real online health consultation communities, recording multi-turn interactions between users and medical professionals or consultation systems. The dataset centers on medical consultation scenarios and includes tasks such as symptom description, disease diagnosis, and health advice, exhibiting strong domain specificity and long-term information dependencies.

C.2 Data Splits

As shown in Table 9, for ShareGPT, to ensure consistent intent decomposition and checklist-based evaluation, we apply a data filtering step that retains multi-turn conversations with moderate lengths and coherent intent flows, while removing instances involving sensitive or unsafe content that may trigger API-level safety refusals. Based on the filtered ShareGPT data, we further construct

[6] `https://sharegpt.com`

ShareGPT-LF by isolating conversations from the legal and financial domains to evaluate cross-domain generalization, and randomly split the remaining data into training and test sets. For JDDC and MedDG, we randomly sample conversation instances and split them into training and test sets.

Table 9: Statistics of conversation-level data splits. All splits are performed at the conversation level.

Dataset	# Training Conversations	# Test Conversations	Avg. # Checklist Items
ShareGPT	750	90	6.70
JDDC	750	400	5.01
MedDG	750	400	4.89
ShareGPT-LF	-	66	6.58

Considering computational cost, our dataset scale is aligned with prior work on multi-turn conversation evaluation [11, 16, 43], and is sufficient to provide a reliable assessment of our method's effectiveness.

C.3 Topic Distribution of ShareGPT

Table 10: Coarse-grained topic distribution of the ShareGPT dataset.

Topic	Programming	Science	Health	Technology	Culture	Law	Finance	Others
Train	419	187	23	23	20	0	0	78
Test	48	21	6	4	2	0	0	9
ShareGPT-LF	0	0	0	0	0	22	44	0

We analyze the topic distribution of ShareGPT using coarse-grained categories. As shown in Table 10, the dataset is dominated by programming and science related conversations, while also covering a diverse range of topics such as health, technology, and culture. The topic distributions of the training and test splits are broadly consistent.

C.4 User Profiling and Checklist Construction

For each conversation instance in the dataset, we prompt Qwen3-Max to extract a structured user profile and a corresponding checklist. Representative examples are shown in Tables 12, 13, and 14. The extracted user profiles and checklists are designed to capture conversation-level user intent and evaluation criteria, while allowing for domain-specific variations across datasets.

For **ShareGPT**, the user profile represents an intent chain with explicit triggering logic, capturing the deterministic progression of user interests, while the checklist focuses on content correctness and coverage, adherence to formatting and instruction constraints, and overall conversation coherence. For **JDDC**, the user profile includes user behavior patterns, contextual background, and explicit goals in the customer-service setting, and the checklist emphasizes effective user guidance, correctness of information delivery, and high-quality emotional support and need fulfillment. For **MedDG**, the user profile captures the chief complaint, detailed symptom descriptions, treatment history, and basic consultation goals, whereas the checklist evaluates proper information elicitation and clarification, correctness of medication and examination recommendations, accuracy of disease assessment, and the provision of appropriate health advice and patient reassurance.

We formalize each conversation task as a tuple (p_{usr}, C, x_1), where p_{usr} denotes the extracted user profile, C the checklist, and x_1 the initial user query of the conversation instance. For JDDC, we additionally extract domain-specific customer-service prior knowledge and provide it to all candidate models to ensure alignment on required business rules.

C.5 Quality and Reliability Validation of LLM-Generated Checklists and User Profiles

To assess the robustness and reliability of our LLM-generated checklists and user profiles, we evaluate checklist quality using four criteria from RLCF [35]: *Naturalness*, *Objectiveness*, *Comprehensiveness*, and *Atomicity*. As shown in Table 11, the extracted checklists achieve consistently high scores across all datasets, indicating the extracted checklists provide a strong foundation for reliable reward modeling. Further, we compare the checklists and user profiles produced by our default extractor,

Qwen3-Max, with those generated by GPT-5.1. The two sets of outputs achieve an average semantic consistency of 91.09%, demonstrating that our extraction pipeline is highly stable and largely independent of the specific LLM used.

Table 11: Reliability and cross-model consistency of LLM-generated checklists and user profiles.

Dataset	Naturalness	Objectiveness	Comprehensiveness	Atomicity	Consistency
ShareGPT	79.10	96.88	68.97	93.75	93.37
JDDC	85.71	82.86	80.00	85.71	89.61
MedDG	90.00	90.00	70.00	83.33	90.29
Avg	85.00	89.91	72.99	87.60	91.09

Table 12: An example conversation from MedDG used for user profiling and checklist construction. The original conversation, user profile, and checklist are in Chinese and are translated into English for presentation, with some content omitted for brevity.

Category	**Content Details**
Dialogue Instance	**User**: Since yesterday, I have been experiencing persistent abdominal pain and intermittent diarrhea, about 7–8 times. Last night my temperature was 37.3°C. What medication should I take? (Female, 30 years old) (从昨天开始到今天肚子一直痛，断断续续拉了7.8次大便，昨晚测量体温37.3度，请问医生我该吃点什么药?（女，30岁）) **Doctor**: Have you eaten any contaminated food, greasy or spicy food, or consumed alcohol? Do you smoke, feel stressed, or have poor sleep? Have you undergone any medical examinations or taken any medication? Acute gastroenteritis is a possible cause. You may take montmorillonite powder and levofloxacin capsules. (你好，请问你是吃不干净的东西了吗? 有没有油腻食物，辛辣刺激性食物，喝酒抽烟，精神压力大，睡眠不好之类有没有做过什么其他检查，有吃过什么药物吗。考虑急性肠胃炎可能性比较大。可以吃蒙脱石散，左氧氟沙星胶囊。) **User**: I ate some greasy food a few days ago. I have not had any examinations recently and have not taken any medication. (前几天吃了点油腻的食物，近期也没有做什么检查，也没有吃什么药。) **Doctor**: That is fine. Both medications are available at pharmacies. I recommend having a stool routine test. (嗯呢。可以的。这两种药药店都有。建议查一个大便常规。) **User**: Okay, thank you. (好的，谢谢。) **Doctor**: Good health relies on self-care rather than only doctors or medications. Sunlight, a positive mindset, and reducing stress are also important. (健康，是靠自己的养护，不是靠医生，和药物，阳光，微笑，就是健康，别给自己压力太大，加油哦。) **User**: Hello doctor, my spouse still has diarrhea after taking the medication, with abdominal pain and diarrhea shortly after meals. Should we continue the medication and observe? ("医生你好，我爱人吃了药后，还是拉稀，和刚吃完饭后，肚子痛，又闹肚子，这个还要接着吃药，看看吗。) **Doctor**: Have you had a stool test or blood test? I suggest going to the emergency department for further evaluation. We need to check for possible electrolyte imbalance and abnormal blood counts. Further treatment may be required, and it is difficult to assess without examinations. (有没有做大便检查。抽血了吗。建议你去急诊看看。观察有没有电解质紊乱和血象增高。是需要进一步治疗的。不化验不好说。)
User Profile	**Demographics**: Female, 30 years old. **Chief Complaint**: Abdominal pain and diarrhea. **Symptom Details**: (1) *Abdominal pain*: Located in the lower abdomen; cramping in nature. (2) *Diarrhea*: 7–8 episodes; watery stool; no blood observed. (3) *Fever*: Body temperature of 37.3°C; duration within one day. **Suspected Trigger**: Consumption of greasy food a few days prior. **Medical History**: None reported. **Current Medication**: None. **User Goals**: (1) Symptom assessment; (2) Medication recommendation; (3) Examination recommendation; (4) Diagnosis clarification.
Checklist	(1) Whether the doctor identifies the patient's key symptoms, including abdominal pain, diarrhea, and fever. (2) Whether the doctor recommends appropriate medications, such as montmorillonite powder and levofloxacin capsules, for treating diarrhea and abdominal pain. (3) Whether the doctor advises relevant examinations, such as a stool routine test. (4) Whether the doctor determines the underlying condition to be acute gastroenteritis.

Table 13: An example conversation from JDDC used for user profiling and checklist construction. The original conversation, user profile, and checklist are in Chinese and are translated into English for presentation, with some content omitted for brevity.

Category	Content Details
Dialogue Instance	**User**: Can you help me change the delivery address for my order? (可以帮我改下订单的地址吗？) **Customer Service**: If the address is within the same city, you can contact the courier to update it. (同一市内可以联系配送员直接修改的哦。) **User**: It's not the same city. The current address is in Shanghai, but I'll be returning to Anhui tomorrow. (不在同一个城市，现在地址是上海，但是我明天要回安徽。) **Customer Service**: Sorry, cross-city address changes are not supported. We recommend placing a new order instead. (抱歉，地址在不同城市不能操作的，只能建议您重新下单哦。) **User**: If I cancel the order, how long will it take to receive the refund? (那我取消订单的话退款多久到账呢？) **Customer Service**: Refund times depend on the payment method: one business day for digital wallets, 1–7 business days for debit cards, and 1–15 business days for credit cards. (微信零钱1个工作日内到账，储蓄卡1-7个工作日内到账，信用卡1-15个工作日内到账的哦！) **User**: Why can't the address be changed? This is really inconvenient. (为什么不能改地址，你们这也太不方便了。) **Customer Service**: We apologize for the inconvenience. Our logistics system still has limitations. (非常抱歉，我们物流还有待完善呢。) **User**: This is quite troublesome—I need the item urgently. (这也太麻烦了，我还急着用呢。) **Customer Service**: We understand your urgency and will continue to improve our service. (非常抱歉！如果是我的话我也会很着急的，我们会改进的！) **User**: Alright. (行吧。) **Customer Service**: Thank you for your understanding. Is there anything else we can help with? (谢谢您的理解！还有什么能帮到您的吗？)
User Profile	**Behavior Patterns**: Attempts to change the delivery address across cities; expresses urgency and dissatisfaction; proactively inquires about refund timing. **General Context**: Currently in Shanghai and returning to Anhui the next day; has an urgent need for the item. **User Goals**: (1) Change the delivery address from Shanghai to Anhui. (2) Inquire about the refund processing time if the order is canceled.
Checklist	(1) Whether the agent identifies or confirms the location involved in the address change request. (2) Whether the agent clearly explains that delivery addresses can only be modified within the same city and explicitly states that cross-city address changes are not supported. (3) Whether the agent acknowledges and soothes the user's urgency and dissatisfaction. (4) Whether the agent suggests placing a new order as an alternative solution. (5) Whether the agent provides accurate refund processing times for different payment methods.

Table 14: An example conversation from ShareGPT used for user profiling and checklist construction. Some content is omitted for brevity.

Category	Content Details
Dialogue Instance	**User**: What are the classic cocktails that every bartender should know how to make and what are their ingredients? **Assistant**: 1. Old Fashioned: Ingredients - bourbon or rye whiskey, sugar, bitters, ice, orange twist or cherry for garnish. 2. Martini: Ingredients - gin or vodka, dry vermouth, ice, olives or lemon twist ... **User**: I've had all of these classic cocktails, but I'm looking for something new to try. Do you have any recommendations? **Assistant**: Sure, here are a few modern cocktail recommendations to try... **User**: The Elderflower Sour sounds interesting. I love trying new cocktails with unique ingredients. **Assistant**: The Elderflower Sour is a great cocktail... **User**: Definitely going to try the Elderflower Sour tonight! Do you have any other unique cocktail recommendations with interesting flavors? I'm in the mood to experiment. **Assistant**: Sure, here are a few more unique cocktail recommendations... **User**: The Smoked Maple Old Fashioned sounds intriguing. How exactly do you use a smoking gun in a cocktail? **Assistant**: To use a smoking gun in a cocktail like the Smoked Maple Old Fashioned... **User**: Wow, I never knew you could use a smoking gun in cocktails. That's a really cool technique! Do you have any other creative cocktail techniques that I should know about? **Assistant**: Yes, there are several creative cocktail techniques... **User**: Wow, these techniques are really cool! I've never heard of fat-washing before. Do you have any tips for getting the proportions right? **Assistant**: Fat-washing is a fun technique...
User Profile	**User Intent Chain with Trigger Logic**: (I1) Request classic cocktails with ingredients and preparation methods; triggered initially when the user seeks foundational cocktail knowledge and re-triggered if recipes are insufficient. (I2) Request modern cocktail recommendations with recipes; triggered after (I1) is fulfilled and re-triggered if preparation details are incomplete. (I3) Request unique or creative cocktail recipes with distinctive flavors; triggered when the user expresses interest in experimentation and re-triggered if diversity is lacking. (I4) Request creative cocktail-making techniques; triggered when the user shows interest in how creative cocktails are prepared and re-triggered if key techniques are omitted.
Checklist	(1) Whether the model's recommended classic cocktails include Old Fashioned and Bloody Mary. (2) Whether the model's recommended modern cocktails include Elderflower Sour. (3) Whether the model's output includes at least 5 classic cocktails. (4) Whether the model's output includes at least 5 modern cocktails. (5) Whether the model provides the recipe and basic preparation method for each cocktail. (6) Whether the model avoids repeating cocktails or other information already provided. (7) Whether the model avoids self-contradictions across multiple turns.

D Details of Candidate LLMs

D.1 Implementation of Candidate LLMs

For open-source models, we obtain the model weights from the HuggingFace[7] and deploy them locally using the vLLM framework [18]. For closed-source models, we invoke the APIs of Qwen3-Max[8], DeepSeek-Chat-V3.2[9], and GPT-5.1[10] through their respective official platforms. Since our study focuses on conversation flow control and task fulfillment rather than complex long-chain reasoning, all candidate models are configured in non-reasoning mode, with the generation temperature set to 0 to reduce randomness and ensure stable and comparable responses across models.

D.2 Pricing of Closed-Source LLMs

Pricing information for closed-source models is obtained from the official billing specifications[11] [12] [13] of their respective providers, as shown in Table 15. In general, the input cost under cache hits

[7] https://huggingface.co

[8] https://bailian.console.aliyun.com

[9] https://platform.deepseek.com

[10] https://platform.openai.com/docs/models/gpt-5.1

[11] https://www.alibabacloud.com/help/en/model-studio/model-pricing

[12] https://api-docs.deepseek.com/quick_start/pricing

[13] https://platform.openai.com/docs/pricing

is substantially lower than the standard input cost. Among the closed-source models considered, DeepSeek-Chat-V3.2 exhibits significantly lower input and output costs than the other models. Note that the cost weight λ in the reward function is defined with respect to the per-token cost.

Table 15: Pricing of closed-source language models.

Model	Input ($/1M tokens)	Cached Input ($/1M tokens)	Output ($/1M tokens)
Qwen3-Max	0.459	0.0918	1.836
DeepSeek-Chat-V3.2	0.28	0.028	0.42
GPT-5.1	1.25	0.125	10

D.3 Considerations on LLM Candidate Set Selection

Our experiments validate the effectiveness of ConvoRouter under both homogeneous (Table 1) and heterogeneous (Table 2, 6) LLM settings. Our homogeneous candidate set reflects prevalent industrial deployment scenarios. In real-world multi-turn conversation systems, LLM service providers (e.g., OpenAI, Alibaba, Anthropic, Google) typically optimize performance and cost within their proprietary LLM families (e.g., OpenAI's GPT series [1] such as GPT-4, GPT-4o, and GPT-4o-mini; Alibaba's Qwen series [37] such as Qwen-7B, Qwen-14B, Qwen-Flash, and Qwen-Max), rather than routing traffic across different LLM families. Moreover, in enterprise private deployments, same-family LLM usage is a standard practice to ensure consistent data formats and stable performance across multi-turn conversations. Therefore, intra-family switching accurately reflects standard industrial routing paradigms and real-world applications. In addition, experiments on heterogeneous LLM sets (Table 2, 6) demonstrate that ConvoRouter maintains strong performance in cross-family settings.

E Training and Inference Time and GPU Resources of ConvoRouter

Table 16: Time and GPU memory consumption for MCTS, training, and inference.

	MCTS	Training	Inference
Time	3h 40min (750 conversations)	1.3 s / batch	0.03 s / decision
GPU Memory	–	18.9 GB	3636 MiB

Training data are generated by running MCTS in parallel across conversation instances. Processing 750 training conversations for a single dataset takes approximately 3 hours and 40 minutes. All experiments are conducted on a single NVIDIA RTX 4090D GPU. ConvoRouter contains 472.9M parameters in total, with 44.42M trainable parameters. ConvoRouter is trained offline using the collected data. During training with a batch size of 32, the peak GPU memory usage is 18.9 GB, and the training throughput is approximately 1.3 seconds per batch. At inference time, routing decisions are made online, requiring 3636 MiB of GPU memory and only 0.03 seconds per decision (including a lightweight retrieval step taking approximately 0.01 seconds). Overall, ConvoRouter is lightweight and computationally efficient.

F Additional Experiments on Candidate LLM Pools

F.1 Effectiveness of Candidate Set Composition

Table 17: Impact of candidate set composition.

Setup	Full Mixed Set	Only Large Models	Δ
SR	88.32	86.84	-1.48

To verify the effectiveness of candidate set composition, we retrained and evaluated ConvoRouter on ShareGPT using a candidate pool consisting exclusively of large models (Qwen3-14B and Llama3.1-8B-Instruct). As shown in Table 17, when restricted to these two large models, ConvoRouter's success

rate drops by 1.48% compared to the full mixed candidate set. This result indicates that models of different scales are not simply interchangeable, but instead exhibit important complementarity in multi-turn conversations. The performance drop suggests that smaller models are beneficial in certain turns, and ConvoRouter effectively leverages this complementarity through routing decisions.

F.2 Evaluation on a Larger Heterogeneous Candidate Pool

Table 18: Performance comparison on ShareGPT using a heterogeneous candidate pool of nine LLMs.

Method	KNN Router	Avengers	ConvoRouter (Ours)
SR	87.39	87.56	91.84

We further evaluate ConvoRouter on ShareGPT using a larger heterogeneous candidate pool of nine LLMs, including Qwen3-4B/8B/14B, Llama3.2-1B/3B, Llama3.1-8B, Qwen3-Max, DeepSeek-Chat, and GPT-5.1. As shown in Table 18, ConvoRouter outperforms KNN Router and Avengers by 4.45% and 4.28%, respectively, further demonstrating its effectiveness in routing across LLMs from different families and scales.

G Robustness to Potential Bias from LLM Choices in Evaluation and MCTS Guidance

Table 19: Task success rates evaluated by two independent evaluators (Qwen3-Max and GPT-5.1), with Spearman's ρ indicating consistent method rankings across domains. GPT-5.1* denotes the variant where GPT-5.1 is used to guide the MCTS search process, and the router is subsequently retrained using the resulting trajectories.

Method	ShareGPT ($\rho = 0.9, 0.9$)			JDDC ($\rho = 0.7$)		MedDG ($\rho = 0.9$)	
	Qwen3-Max	GPT-5.1	GPT-5.1*	Qwen3-Max	GPT-5.1	Qwen3-Max	GPT-5.1
Qwen3-14B	83.24	77.77	77.77	76.41	78.52	71.38	69.50
KNN Router [17]	83.37	76.45	77.52	77.78	78.71	70.95	69.23
Avengers [42]	83.93	78.29	78.01	77.89	78.37	70.98	70.35
ConvoRouter (Ours)	87.46	81.01	82.87	83.06	83.98	78.42	76.55
MCTS Router	94.11	87.00	87.62	88.26	89.02	86.18	83.17

G.1 Consistency Across Different LLMs Used as Final Evaluators

To assess robustness to evaluator choice and potential preference bias, we evaluate methods using two independent evaluators, Qwen3-Max and GPT-5.1. As shown in Table 19, method rankings remain largely consistent across all three conversation domains, with high Spearman's ρ values. ConvoRouter consistently outperforms single LLMs and routing baselines under both evaluators, indicating that its performance gains are robust to evaluator-specific preferences.

G.2 Consistency Across Different LLMs Used for MCTS Guidance

In addition, to further verify that ConvoRouter does not merely learn the implicit preferences of Qwen3-Max, we introduce an additional GPT-5.1* setting, where GPT-5.1 replaces Qwen3-Max to guide the MCTS search process, and the router is retrained on the resulting search trajectories. Results show that under the GPT-5.1* setting, ConvoRouter still achieves a success rate of 82.87%, outperforming all baselines by at least 4.86%. This demonstrates that ConvoRouter's performance gains do not depend on a particular MCTS guiding model. Regardless of which LLM is used for search, its improvements remain consistent, indicating that the learned routing policy is robust to LLM-specific biases introduced during the search process.

H Human Evaluation

We conduct a human evaluation on the ShareGPT test set with nine volunteers (seven male and two female) from our institution, each evaluating 30 samples. For each sample, annotators were shown two complete multi-turn conversations, one generated by ConvoRouter and the other by a baseline, presented in randomized order with method identities hidden. Following a unified set of evaluation guidelines, annotators were instructed to assess the conversations across six aspects: task completion and practical helpfulness, correctness and reliability, understanding and adaptation to user needs, multi-turn coherence and context retention, clarity and relevance, and overall interaction quality. They then determined which conversation was better overall or marked a tie.

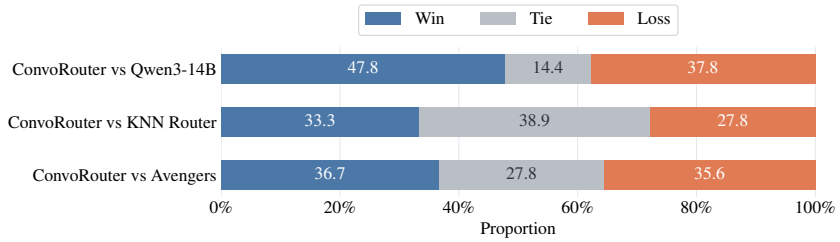


Figure 4: Human evaluation results on ShareGPT, showing the win, tie, and loss rates of ConvoRouter against three baselines.

As shown in Figure 4, ConvoRouter achieves an average win rate of 39.3%, compared with an average loss rate of 33.7% and a tie rate of 27.0% across three representative baselines. The human evaluation results are broadly consistent with the automatic evaluation results, further supporting ConvoRouter's advantage in overall multi-turn dialogue quality and its potential to improve real-world user experience.

I Details of Cost-Aware Routing

I.1 Statistical Significance and Cost Analysis

By comparing the results before and after introducing the cost-aware reward to the closed-source candidate set on ShareGPT, we constructed 95% confidence intervals (CIs) for the paired differences using a t-distribution. The average success rate drop is 1.4% (95% CI: [-1.5%, 4.3%]). Since the interval includes zero, this performance variance is statistically insignificant. Conversely, the cost drops by 0.35 (95% CI: [0.30, 0.41]), indicating a statistically significant reduction. These findings rigorously confirm that ConvoRouter significantly cuts costs while maintaining stable performance.

I.2 Cost and Performance under Varying Cost Weights λ

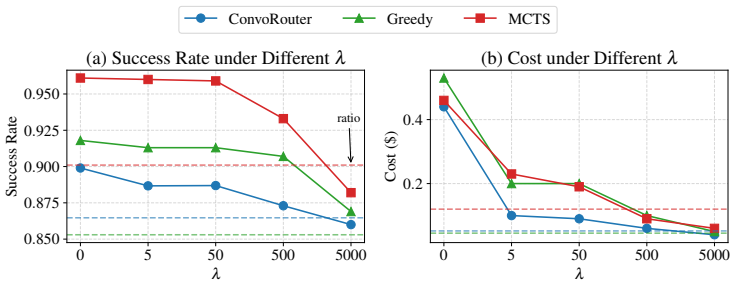


Figure 5: SR and cost of ConvoRouter, Greedy Router, and MCTS Router under different values of the cost weight λ. Dashed horizontal lines correspond to the ratio-based reward setting.

As shown in Figure 5, we analyze the SR and cost of ConvoRouter, Greedy Router, and MCTS Router under different λ. Overall, increasing λ consistently reduces cost for all methods, accompanied

by a moderate decline in SR. When λ increases from 0 to 5, cost is substantially reduced with only a slight SR drop, indicating that a small λ can effectively lower inference cost with minimal performance impact. As λ increases from 5 to 50, both SR and cost become relatively insensitive to λ, corresponding to a stable performance-cost trade-off regime. Across all λ, ConvoRouter consistently incurs lower cost than MCTS, which can be attributed to behavior cloning bias toward high-frequency low-cost actions in MCTS trajectories, underrepresenting infrequent yet performance-critical high-cost decisions; this results in lower overall cost with a SR degradation. We further evaluate a ratio-based reward $\mathcal{R} = \frac{R_t - R_{t-1}}{R_t^{\text{cost}}}$, shown as dashed lines in Figure 5; under this formulation, Greedy Router almost always selects DeepSeek due to its dominant low price, while both MCTS Router and ConvoRouter also exhibit performance degradation, indicating that ratio-based rewards fail to stably capture the performance–cost trade-off in candidate sets with substantial price disparities.

Table 20: Impact of KV cache sharing rate α on routing performance and efficiency.

α	SR↑	Cost↓	Avg Switch
0.0	88.69	0.090	1.53
0.5	88.49	0.080	1.60
1.0	88.47	0.076	1.62

Table 21: Impact of KV cache retention rate η on routing performance and efficiency.

η	SR↑	Cost↓
0.4	88.55	0.120
0.8	88.19	0.112
1.0	88.69	0.090

I.3 Cost Formulation under KV Cache Sharing

To support emerging cross-LLM KV cache sharing, we introduce a sharing rate $\alpha \in [0, 1]$ into the cost formulation in Eq. 6:

$$R_t^{\text{in}} = \begin{cases} -\Big[c_{\text{in}}^{(a_t)}\big((L(\tau_{t-1}) \bmod B) + L(x_t)\big) + c_{\text{cache}}^{(a_t)}\Big\lfloor \frac{L(\tau_{t-1})}{B} \Big\rfloor \cdot B\Big], & a_t = a_{t-1} \\ -\Big[c_{\text{in}}^{(a_t)}\Big(L(\tau_{t-1}) - \Big\lfloor \frac{\alpha L(\tau_{t-1})}{B} \Big\rfloor \cdot B + L(x_t)\Big) + c_{\text{cache}}^{(a_t)}\Big\lfloor \frac{\alpha L(\tau_{t-1})}{B} \Big\rfloor \cdot B\Big], & a_t \neq a_{t-1} \end{cases} \tag{12}$$

The parameter α controls the proportion of historical KV cache that can be reused when switching between different LLMs. Setting $\alpha = 0$ degenerates to the standard setting without cache sharing, while larger values of α enable more aggressive reuse across models.

We simulated routing performance across varying α values. As shown in Table 20, increasing α leads to more KV cache reuse across models, thereby reducing the effective cost of cross-model switching. This encourages more fine-grained routing decisions with slightly higher switching frequency, while maintaining a stable success rate and significantly reducing inference cost.

I.4 Cost Formulation under Imperfect KV Cache Reuse

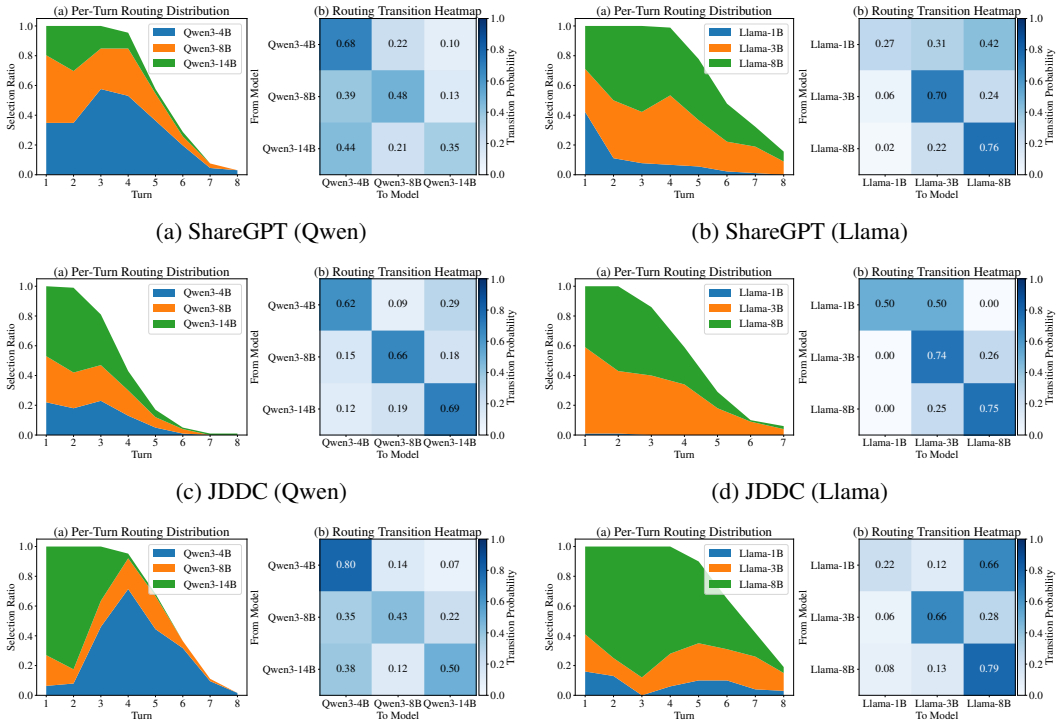
In practice, KV caches may be partially unavailable due to expiration, eviction, or concurrent requests, even when the same LLM is selected in consecutive turns. To account for imperfect cache reuse, we introduce a retention rate $\eta \in [0, 1]$ into Eq. 6, representing the expected proportion of historical context that remains reusable. Unlike the cross-model cache sharing rate α, η characterizes cache availability when the same LLM is reused. The input cost is extended as follows:

$$R_t^{\text{in}} = \begin{cases} -\Big[c_{\text{in}}^{(a_t)}\Big(L(\tau_{t-1}) - \Big\lfloor \frac{\eta L(\tau_{t-1})}{B} \Big\rfloor \cdot B + L(x_t)\Big) + c_{\text{cache}}^{(a_t)}\Big\lfloor \frac{\eta L(\tau_{t-1})}{B} \Big\rfloor \cdot B\Big], & a_t = a_{t-1} \\ -c_{\text{in}}^{(a_t)}\big(L(\tau_{t-1}) + L(x_t)\big), & a_t \neq a_{t-1} \end{cases} \tag{13}$$

Setting $\eta = 1$ recovers the full-cache-hit assumption in Eq. 6, while $\eta = 0$ corresponds to no cache reuse even when the same LLM is selected consecutively.

We evaluate different retention rates under the standard setting without cross-model cache sharing. As shown in Table 21, ConvoRouter maintains a stable success rate across different cache retention rates, while inference cost decreases as η increases. These results demonstrate that ConvoRouter remains effective under imperfect KV cache reuse, with greater cache availability further improving cost efficiency.

J Analysis of ConvoRouter's Routing Behavior



(a) ShareGPT (Qwen) (b) ShareGPT (Llama)

(c) JDDC (Qwen) (d) JDDC (Llama)

(e) MedDG (Qwen) (f) MedDG (Llama)

Figure 6: Per-turn routing selection distributions (stacked proportion plots) and inter-turn routing transition heatmaps for the Qwen (left) and Llama (right) candidate sets across three datasets. The stacked areas show the proportion of conversation trajectories selecting each model at each turn; the total height decreases in later turns as fewer conversations extend to longer horizons.

This section analyzes routing behaviors across different conversation scenarios by visualizing multi-turn LLM routing trajectories. As shown in Figure 6, we present per-turn LLM selection distributions (stacked proportion plots) and inter-turn transition patterns (transition matrices) on three datasets to characterize the dynamic evolution of routing strategies over conversation stages. By comparing different candidate LLM sets within the same dataset, we examine how routing adapts to task contexts and conversation phases, highlighting differences in exploration, convergence, and stability.

On **ShareGPT**, the Qwen candidate set favors medium-sized models in early turns and gradually shifts toward smaller models as the conversation progresses, with the transition matrix indicating frequent fallback transitions from larger to smaller models, resulting in a clear strong-to-weak convergence pattern. In contrast, the Llama candidate set more often maintains or returns to larger models in later turns, with higher self-loop and back-transition probabilities, suggesting sustained reliance on high-capacity models in open-domain multi-turn conversations.

On **JDDC**, the Qwen candidate set quickly converges to large models in the initial turns and remains relatively stable thereafter, with a transition structure dominated by self-loops, indicating limited cross-model switching once a routing strategy is established. By comparison, the Llama candidate set is largely dominated by medium and large models in early turns and exhibits noticeable transitions from medium to large models, reflecting a preference for high-capacity models in task-constrained e-commerce conversation scenarios.

On **MedDG**, the Qwen candidate set strongly prefers large models at the beginning of the conversation, followed by a stage-wise shift toward medium and smaller models in later turns, with frequent transitions from large to medium models, capturing strategy adjustments during information gathering and verification phases in medical conversations. In contrast, the Llama candidate set consistently

concentrates on large models across most turns, with dominant self-loop transitions, indicating sustained reliance on high-capacity models under high-stakes and safety-critical medical settings.

K Case Studies

As shown in Figures 7, 8, and 9, the three case studies cover distinct multi-turn conversation scenarios, including script writing in ShareGPT, medical consultation in MedDG, and e-commerce customer service in JDDC, where user requirements progressively evolve and become more specific as the conversation unfolds. Across these scenarios, ConvoRouter is able to perceive the conversation stage and select models that better match the current requirement structure, thereby maintaining decision consistency across turns and incrementally completing key information. By employing higher-capacity models for high-risk or critical decision stages and more lightweight or stable models for execution details and supplementary explanations, ConvoRouter ultimately achieves full coverage of the checklist items. In contrast, some baseline methods rely on a fixed model or lack stage awareness, leading them to overlook newly introduced constraints or details in later turns and fail to fully satisfy conversation-level objectives, while others switch models without a holistic understanding of conversation stages and risk, resulting in oscillations or drift along critical decision paths.

Overall, these cases consistently demonstrate that success in multi-turn conversation does not depend on producing the strongest possible response at each individual turn, but rather on the router's ability to perceive conversation stages and make long-horizon model selection decisions, thereby jointly ensuring directional correctness, detail completeness, and risk control throughout the evolving interaction and ultimately achieving stable conversation-level goal fulfillment.

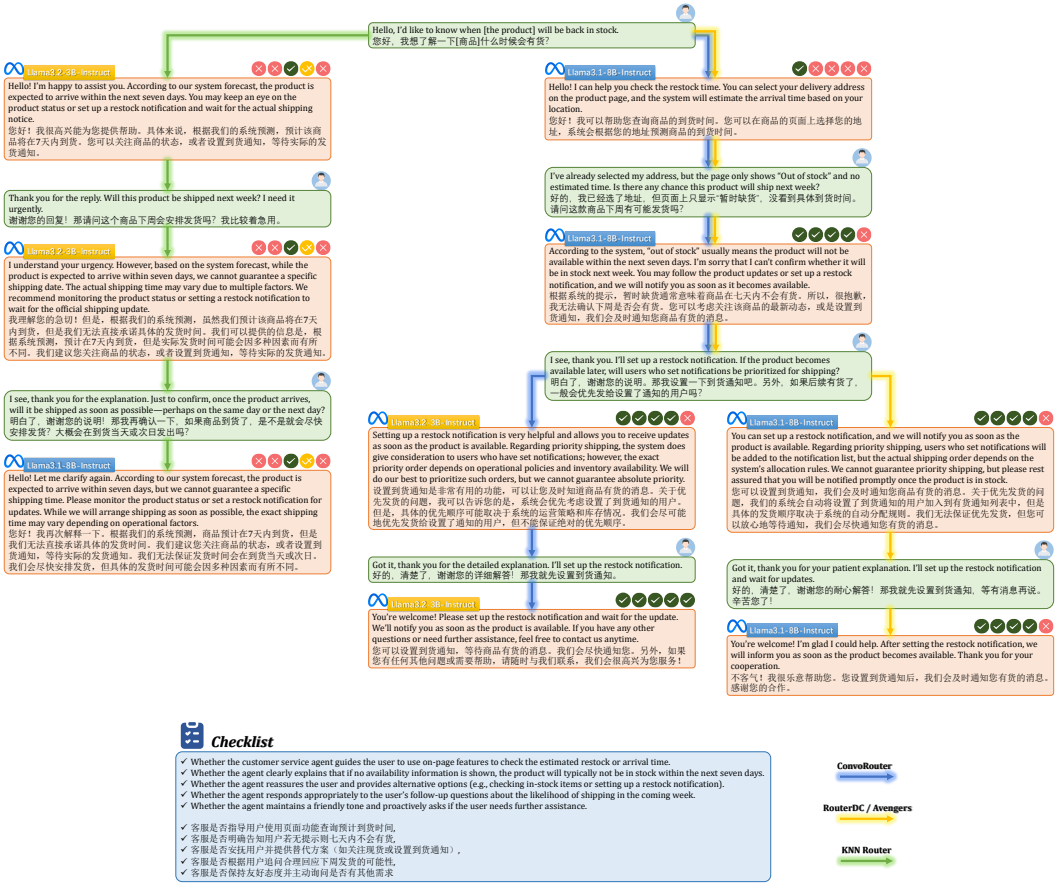


Figure 7: Case study on multi-turn e-commerce customer service in JDDC with Llama candidate set.

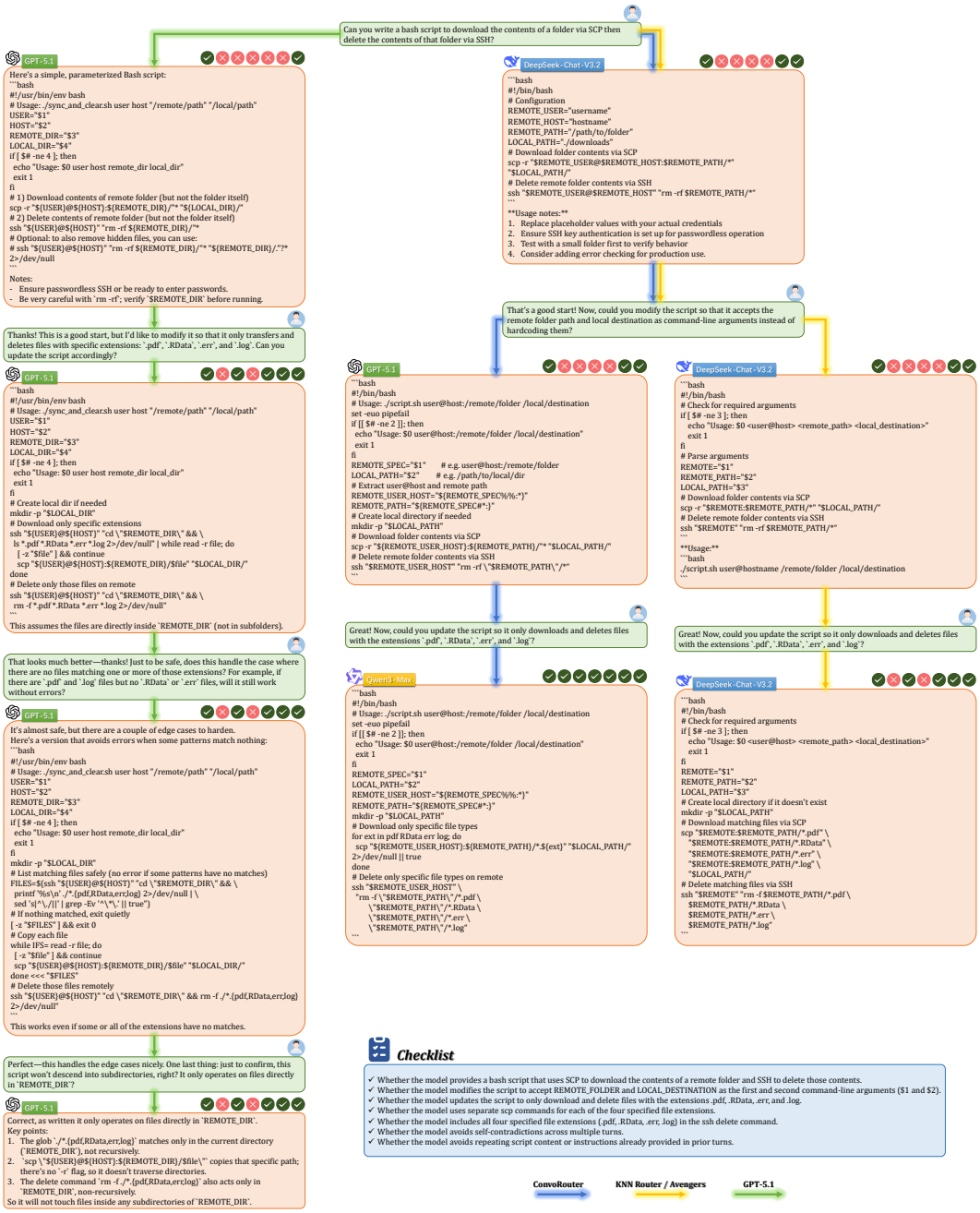


Figure 8: Case study on multi-turn script writing in ShareGPT with closed-source LLM candidate set.

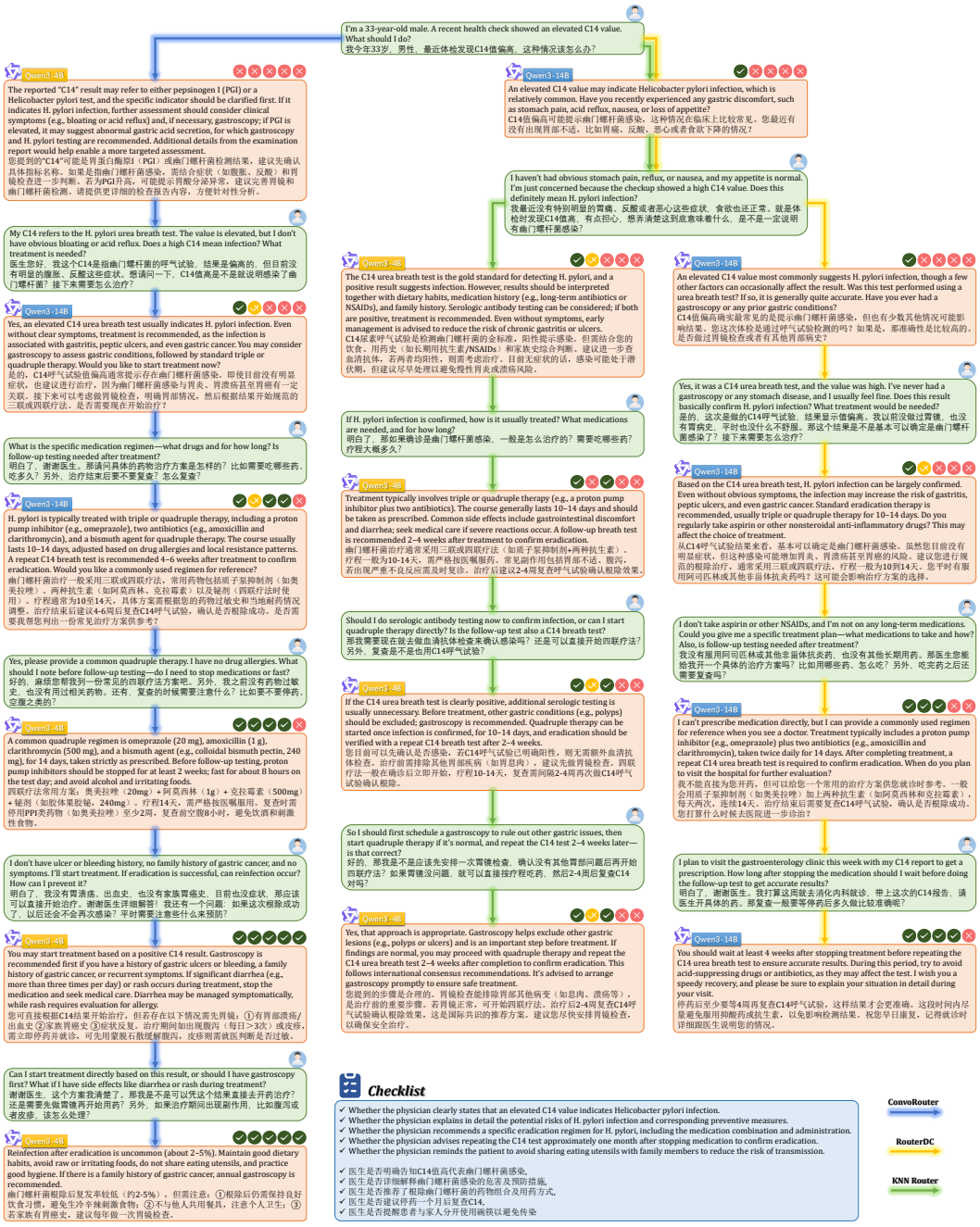


Figure 9: Case study on multi-turn medical consultation in MedDG with Qwen candidate set.

L Prompting Details

We use Qwen3-Max as both the user simulator and the reward model, and design corresponding system prompts for ShareGPT, JDDC, and MedDG, which are instantiated with the user profile p_{usr} and the checklist C extracted from each conversation instance. The user simulator and evaluator of open-domain ShareGPT are not tied to a specific domain, enabling a general and reusable simulation and evaluation for new domains. The detailed system prompts are illustrated in Figures 10–15.

System Prompt for the ShareGPT User Simulator

You are a human user engaging in a multi-turn conversation with an AI assistant. Please generate user utterances that are natural, coherent, and aligned with the specified conversational goals.

Conversation Goals
- User intents: [User Intents]

Please follow the rules below throughout the conversation:
1. In each turn, your utterance should be guided by the currently triggered intent. Do not introduce goals that are unrelated to the specified intents or deviate from the conversation scope.
2. Your response should naturally follow the assistant's previous reply and maintain conversational coherence. You may exhibit realistic user behaviors such as asking follow-up questions, seeking clarification, or restating information when appropriate.
3. Use natural, conversational language rather than structured or list-style expressions, and keep each utterance at a moderate length.
4. If all intents have been fully satisfied, or no further intent can be reasonably triggered, terminate the conversation by outputting: <End of Conversation>.

Figure 10: System prompt for the ShareGPT user simulator.

System Prompt for the ShareGPT Reward Model

You are a professional dialogue evaluator. Your task is to assess the assistant's performance in the conversation based on a given checklist and determine whether the dialogue has terminated.

Dialogue History
[Dialogue History]

Checklist
[Checklist]

Scoring Guidelines
For each checklist item, evaluate whether the assistant has completed the corresponding task behavior in the dialogue and assign a score according to the following criteria:
- 1: The assistant clearly and fully satisfies the item, providing complete and explicit information aligned with the user's intent;
- 0.5: The assistant partially satisfies the item, but the response is vague, incomplete, or only indirectly addresses the requirement;
- 0: The assistant does not address the item at all or clearly fails to fulfill the requirement.
All judgments must be strictly based on the observed dialogue content and the behaviors specified in the checklist. Do not speculate, infer, or rationalize any assistant behavior that is not explicitly present in the dialogue.

Dialogue Termination Determination
The dialogue is considered terminated, and the field 'done' should be set to 1, if and only if the user's final complete utterance exactly matches the string "<End of Conversation>". Otherwise, set 'done' to 0.

Output Format
Return only the following JSON object, without any additional text or explanation:
{
"checklist": [score_1, score_2, ..., score_n],
"done": 0 or 1
}

Figure 11: System prompt for the ShareGPT reward model.

System Prompt for the JDDC User Simulator (English)

You are an e-commerce platform user. Based on the given user background and goals, please engage in a multi-turn conversation with the customer-service system.

User Behavioral Patterns
During the conversation, please consistently exhibit the following behavioral patterns:
- Behavioral patterns: [Behavior Patterns]

User General Context
- Background information: [General Context]

User Goals
- Dialogue goals: [User Goals]

Please strictly follow the dialogue rules below:

1. You may only act as the user. Do not simulate, replace, or guide the customer-service agent's responses.
2. In each turn, generate a natural user utterance based on the customer-service agent's previous reply, using language and tone consistent with real e-commerce interactions.
3. Your utterances should remain focused on the specified user goals. Do not introduce unrelated topics or requests.
4. Once you believe that sufficient information has been obtained to fully satisfy the user goals, you may directly terminate the conversation without waiting for additional confirmation or actions.
5. If the customer-service agent explicitly indicates that the request cannot be further processed, or you determine that all goals have been completed, output: <End of Conversation>.

System Prompt for the JDDC User Simulator (Chinese)

你是一名电商平台用户，请根据给定的用户背景和目标，与客服系统展开多轮对话。

用户行为特征
在对话过程中，请始终体现以下行为倾向：
- 行为特征：[用户行为特征]

用户背景
- 背景信息：[用户通用背景]

用户目标
- 对话目标：[用户目标]

请你严格遵循以下对话规则：

1. 你只能扮演用户角色，不要模拟、代替或引导客服的发言。
2. 在每一轮对话中，请基于客服上一轮的回复内容，自然地生成当前轮的用户发言，语气应符合真实电商用户的交流习惯。
3. 你的发言应始终围绕用户目标展开，不要主动引入与目标无关的话题或需求。
4. 当你认为已获取到满足目标的完整信息时，可以直接结束对话，无需等待额外确认或操作。
5. 如果客服明确表示无法继续处理你的请求，或你判断所有目标均已完成，请输出：<结束对话>。

Figure 12: System prompt for the JDDC user simulator.

System Prompt for the JDDC Reward Model (English)

You are a professional customer-service dialogue evaluator. Your task is to assess the customer-service agent's behavior throughout the conversation based on the given checklist and to determine whether the dialogue has ended.

Dialogue History
[Dialogue History]

Checklist
[Checklist]

Scoring Guidelines
For each item in the checklist, evaluate the customer-service agent's behavior in the dialogue and assign a score according to the following guidelines:
- 1: The agent clearly and fully completed the corresponding service behavior;
- 0.5: The agent partially completed the service behavior, but the response is unclear, insufficient, or only indirectly addresses the requirement;
- 0: The agent did not address the service behavior at all or clearly failed to fulfill the requirement.
When assigning scores, strictly base your judgment on the content explicitly present in the dialogue. Do not rely on common sense, assumptions, or speculative reasoning to fill in missing information.

Dialogue Termination Determination
The dialogue is considered terminated if and only if the user's final complete utterance exactly matches the string "<End of Conversation>". In this case, set 'done' to 1; otherwise, set 'done' to 0.

Output Format
Return the evaluation result strictly in the following JSON format, without any additional text or explanation:
{
"checklist": [score_1, score_2, ...],
"done": 0 or 1
}

System Prompt for the JDDC Reward Model (Chinese)

你是一名专业的客服对话评价器。你的任务是依据给定的 Checklist，对客服在整个对话过程中的行为进行逐项评估，并判断该对话是否已经结束。

对话历史
[对话历史]

Checklist
[Checklist]

评分标准
请针对 Checklist 中的每一项内容，结合对话历史进行判断，并给出对应评分：
- 1：客服明确且充分地完成了该项服务行为；
- 0.5：客服部分完成了该项服务行为，但表达不清晰、不充分，或仅以间接方式涉及；
- 0：客服完全未涉及该项服务行为，或明显未完成该项要求。
评分时请严格依据对话中实际出现的内容进行判断，不得基于常识或主观推断对缺失信息进行合理化补全。

对话终止判定
若且仅若用户在最后一轮对话中的完整回复严格等于字符串"<结束对话>"，则认为该对话已经结束，并将 done 设为 1；否则，将 done 设为 0。

输出格式
请仅按照以下 JSON 格式输出评价结果，不得包含任何额外文本或解释说明：
{
"checklist": [评分1, 评分2, . . .],
"done": 0 或 1
}

Figure 13: System prompt for the JDDC reward model.

System Prompt for the MedDG User Simulator (English)

You are a patient participating in an online medical consultation. Based on the given patient background and consultation goals, please engage in a multi-turn conversation with the medical system.

Patient Information
- Age: [Age]
- Gender: [Gender]
- Chief complaint: [Chief Complaint]
- Trigger or onset: [Suspected Trigger]
- Medical history: [Medical History]
- Current medication: [Current Medication]

Symptom Details
- Detailed description of current symptoms: Symptom Details

Consultation Goals
- Consultation goals: [User Goals]

Please strictly follow the dialogue rules below:

1. You may only act as the patient. Do not simulate or replace the doctor's role.
2. In each turn, generate a natural, realistic, and logically coherent patient utterance based on the doctor's previous response.
3. You should exhibit a help-seeking and awaiting-professional-judgment patient stance. Do not proactively correct, confirm, or explain the doctor's medical conclusions.
4. Your utterances should remain focused on the consultation goals. Do not introduce topics unrelated to the goals or deviate from the current consultation theme.
5. You may terminate the conversation only when you are confident that the consultation goals have been sufficiently satisfied, or when the doctor explicitly indicates that no further assistance can be provided. Before deciding to end the conversation, please ensure that: The consultation goals have been clearly and accurately addressed by the doctor; The doctor's advice is explicit and actionable; No remaining questions or ambiguities require further clarification.
6. When you determine that the conversation has met the termination conditions, output: <End of Conversation>.

System Prompt for the MedDG User Simulator (Chinese)

你是一名正在进行在线问诊的患者，请根据给定的病人背景和就诊目标，与医生系统展开多轮对话。

患者基本信息
- 年龄：[年龄]
- 性别：[性别]
- 主诉：[主要不适或症状]
- 诱因或起病情况：[可能的诱发因素]
- 既往病史：[既往相关病史]
- 当前用药情况：[是否正在用药及用药情况]

症状细节
- 当前症状的具体描述：[症状细节描述]

就诊目标
- 就诊目标：[希望通过本次问诊获得的信息或建议]

请你严格遵循以下对话规则：

1. 你只能扮演患者，不得模拟或代替医生角色。
2. 在每一轮对话中，请基于医生上一轮的回复内容，生成自然、真实且具有逻辑一致性的患者发言。
3. 你应表现出寻求帮助、等待专业判断的患者状态，不要主动纠正、确认或解释医生的专业结论。
4. 你的发言应始终围绕就诊目标展开，不要主动引入与目标无关的问题，也不要偏离当前就诊主题。
5. 只有在你确认就诊目标已被充分满足，或医生明确表示无法继续提供帮助时，才可以结束对话。在判断是否结束前，请确认：就诊目标是否已被医生清晰、准确地回应；医生给出的建议是否明确且具有可操作性；是否仍存在需要进一步澄清的问题。
6. 当你确认对话已满足结束条件时，你的发言应为：<结束对话>。

Figure 14: System prompt for the MedDG user simulator.

System Prompt for the MedDG Reward Model (English)

You are a professional medical dialogue evaluator. Your task is to assess the doctor's performance throughout the medical consultation based on the given dialogue history and checklist, and to determine whether the consultation has ended.

Dialogue History
[Dialogue History]

Checklist
[Checklist]

Scoring Guidelines
Based on the dialogue history, evaluate whether the doctor has completed each item in the checklist and assign a score according to the following criteria:
- 1: The doctor has clearly and fully completed the item, with information that is explicit, complete, and medically sound;
- 0.5: The doctor has partially addressed the item, but the response is unclear, insufficient, or only indirectly related;
- 0: The doctor did not address the item at all or clearly failed to fulfill the corresponding consultation responsibility.
When assigning scores, strictly base your judgment on the content explicitly present in the dialogue and the behaviors required by the checklist. Do not make subjective assumptions or rationalize missing information that is not explicitly provided by the doctor.

Dialogue Termination Determination
The consultation is considered terminated if and only if the patient's final complete utterance exactly matches the string "<End of Conversation>". In this case, set 'done' to 1; otherwise, set 'done' to 0.

Output Format
Return the evaluation result strictly in the following JSON format, without any additional text or explanation:
{
"checklist": [score_1, score_2, ...],
"done": 0 or 1
}

System Prompt for the MedDG Reward Model (Chinese)

你是一名专业的医疗问诊对话评价器。你的任务是基于给定的对话历史和 Checklist，逐项评估医生在问诊过程中的表现，并判断问诊是否已经结束。

对话历史
[对话历史]

Checklist
[Checklist]

评分标准
请根据对话历史，逐项判断医生是否完成 Checklist 中的每一项内容，并按照以下标准进行评分：
- 1：医生已明确且充分完成该项内容，信息表达清晰、完整，且在医学上合理；
- 0.5：医生对该项内容有所涉及，但表达不够清晰、信息不充分，或仅作了间接回应；
- 0：医生完全未涉及该项内容，或明显未履行相应的问诊职责。
评分时请严格依据实际对话内容与 Checklist 所要求的服务行为进行判断，不得进行主观推测，也不得合理化补全医生未明确给出的信息。

对话终止判定
若且仅若患者在最后一轮对话中的完整回复严格等于字符串"<结束对话>"，则认为该对话已经结束，并将 done 设为 1；否则，将 done 设为 0。

输出格式
请仅按照以下 JSON 格式输出评价结果，不得包含任何额外文本或解释说明：
{
"checklist": [评分1, 评分2, . . .],
"done": 0 或 1
}

Figure 15: System prompt for the MedDG reward model.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Yes, the paper discusses the limitations of the work performed by the authors.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not include theoretical results.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have provided detailed training settings and datasets information in the Experiments section and Appendix.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We use public dataset to conduct experiments, and our data and source code will be released.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We have specified all the experimental setting and details in the Experiments section and Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Error bars are not reported because it would be too computationally expensive.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiments are conducted on a single NVIDIA RTX 4090D GPU.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes, the broader impacts are discussed in the paper.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: No such risks.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have cited all referenced papers and open-source assets.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We construct data for multi-turn LLM routing using multiple publicly available multi-turn conversation datasets, with details provided in the Experiments section and Appendix.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The human evaluation was conducted by internal volunteers to assess model-generated conversations, without crowdsourcing or studying the annotators themselves.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Not involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Yes, the paper describes the usage of LLMs for LLMs are important component of experiments.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.